METHODOLOGY FOR THE ANALYSIS OF VARIABLE THROUGH-PUT PRODUCTION PROCESSES

Stuart M. Novak

METHODOLOGY FOR THE ANALYSIS OF VARIABLE THROUGHPUT PRODUCTION PROCESSES

by

STUART M. NOVAK
Lieutenant, United States Navy

S.B., United States Naval Academy
(1964)

Submitted in Partial Fulfillment of the

Requirements for the

Master of Science Degree in Ocean Engineering

and the Professional Degree, Naval Engineer

at the

Massachusetts Institute of Technology

June, 1971

METHODOLOGY FOR THE ANALYSIS OF VARIABLE THROUGHPUT PRODUCTION PROCESSES

by

Stuart M. Novak
Lieutenant, United States Navy

Submitted to the Department of Naval Architecture and Marine Engineering in partial fulfillment of the requirements for the degree of Master of Science in Ocean Engineering and the Professional Degree, Naval Engineer.

ABSTRACT

In order to provide more than an intuitive approach to decision making when faced with complex multivariable production processes such as those encountered in shipyards, a methodology for the analysis of such processes is sought.

It is obvious from the literature survey that 1) severely restrictive assumptions underlie all purely theoretical approaches, thus the precise results obtained by theoretical methods are solutions to problems that barely resemble the real world situations under consideration; 2) the application of mathematical optimization techniques is restricted due to the combinatorial nature of shipyard production processes which results in computational infeasibility for relatively small scale systems; 3) computer simulations of queueing models offer the greatest potential to the decision maker since they possess the ability to describe the interaction of stochastically determined processes, while resorting only to the use of relatively uncomplicated building block modules.

Basic building block modules demonstrating real world characteristics such as throughput transport, buffer storage, and stochastic service times for both single and multichannel queues are developed. These rather general building blocks are then modified as necessary and combined to describe a "typical" variable throughput multistaged production process. The resulting computer program, its restrictions and potential uses are noted and recommendations for additional research are made.

Thesis Supervisor: Ernst G. Frankel

Title: Professor of Marine Engineering

ACKNOWLEDGMENTS

First and foremost I wish to sincerely thank Professor E. G. Frankel. It would be unfair, though, to merely thank him for the ideas, guidance and understanding he provided in his role as thesis supervisor, for he also provided educational experiences both formal and informal that I consider invaluable.

I would also like to express my appreciation to Mrs. V. Liddell, whose expert typing abilities and editorial comment were of great assistance.

Finally, I wish to thank my wife, Connie for her patience and understanding not only during the course of this investigation, but also during the last three years while I was studying at M.I.T.

Table of Contents

Page

ABSTRACT 2

ACKNOWLEDGMENTS 3

LIST OF FIGURES AND TABLES 5

I. INTRODUCTION 6

A. Background 6
B. Objectives 13

II. SURVEY OF LITERATURE 16

A. Linear Programming 16
B. Dynamic Programming 20
C. Control Theory 24
D. Queueing Models 27

1. Queueing Theory 27
2. Simulation of Queueing Models 28

E. Evaluation of Previous Methods 37

III. TOWARDS A METHODOLOGY 40

A. Rationale for Simulation of Queueing Model 40

1. Desirable Characteristics of Model 41
2. General vs. Problem Oriented Computer Languages 44
3. Fixed vs. Variable Time Increment Methods.... 46

IV. FORMULATION OF MODEL 49

A. Formulation of Problem 49
B. Collection and Processing of Data 55
C. Formulation of Mathematical Model 58
D. Formulation of Computer Program 60
E. Validation of Computer Program 73
F. Design of Simulation Experiments and Analysis of Simulated Data 73

V. CONCLUSIONS 75

VI. RECOMMENDATIONS 77

APPENDIX A - Variable Definitions .....

APPENDIX B - Generation of Stochastic Variates .....

APPENDIX C - Basic Building Blocks for the Simulation of Production Processes .....

APPENDIX D - Simulation of a Variable Throughput Production System Steel Processing Facility

BIBLIOGRAPHY .....

List of Figures and Tables

Figure		Page
1	Steel Processing Facilities	10
2	Subnetwork of Steel Preprocessing Activities	12
3	Linear Programming Tableau for m=3, n=2 Job Shop.	19
4	Production Control Network	25
5	Generalized Control System	26
6-7	Results of the Simulation of Job Shops Conducted by Nanot	32
8	Module of a "Typical" Process	42
9	Flow Chart for Planning Simulation Experiments	50
10	Schematic Diagram for the Portion of the Steel Processing Facility under Consideration	52
11	Distribution of Plate Sizes	55
12	PMFs for Leveler Service Time	56
13	Subdivision of a Multistaged System into Individual Processes	61
14	The Interfacing of Building Block Modules	62
15	Generalized Computer Flow Chart for a System Comprised of N Processes	69
16	Leveling Process Flow Chart Showing Modifications Necessary to Accommodate a Variable Throughput	71

Table		
1	Elements of Steel Processing Processes	11
2	Queueing Disciplines	31
3	Service Time Parameters for Cutting Process	57

I. INTRODUCTION

A. Background

In recent years a large amount of investment has been put into improving ship production and shipbuilding in the world. Despite these investments, major funding for shipbuilding research, mathematical modeling, logical design of operations, systems design and system integration, to date, has not been allocated. Although some effort has been devoted to the integration of equipment, plant layout and economic analysis, a review of some of the recent investments in ship production processes reveals that a majority of the decisions regarding these expenditures was made rather piecemeal, relying heavily on intuition.

Perhaps a reason that there has been a lack of effort in shipyard production research stems from the very nature of the shipyard production processes themselves. The variety of products (which includes ships, submersibles, and ocean platforms, to name the major items) requires that the various processes possess a wide range of production technology in order to be able to respond to the "custom" orders which they receive. Furthermore, the individual processes experience extremely varying demands which often result in seriously intermittent production flow in various process sequences. Generally the convenience of production lines, that is, a rigidly sequential process flow shop,

does not exist. The varied nature of the throughputs for a given facility, or even for a given machine, are such that one immediately finds himself wrestling with numerous variables, and a seemingly endless number of production alternatives. The problem then appears to be so complex at even the lowest level of consideration, that one is tempted to make the "necessary" simplifying assumptions, rely on easy-to-grasp descriptions such as average service time or maximum rated throughput, and then allow himself to be guided by his intuition based upon personal production experience.

The complex production processes referred to result in what is generally called the job shop. The job shop does not produce for inventory, but instead holds in readiness a flexible producing system that has as its most important asset its flexible production and/or assembly capability (which includes a broad range of labor skills and a sizable capital investment in machinery). Because of the flexibility required by the demands of the individual products, the internal complications of job shop systems are much greater than for any other production systems.

There are six major problem areas associated with the job shop systems. These are:

1) Design and layout of the facility;

2) Forecasting of demand;

3) Aggregate planning for the use of the facilities;
4) Scheduling and control of orders, labor and equipment;
5) Procurement of materials; and
6) Establishing a bidding policy to achieve a balance between the use of labor and the facilities on hand.

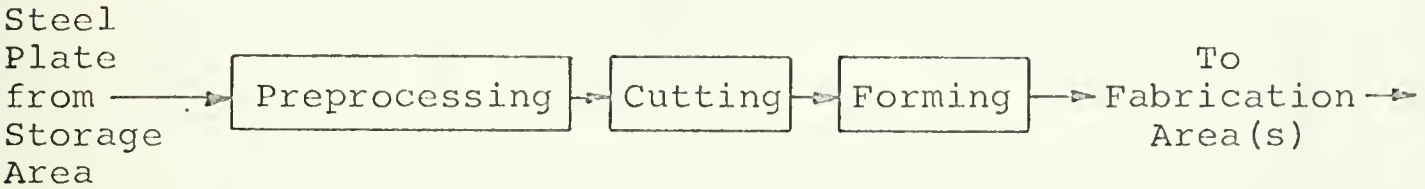
In order to analyze in detail the job shop system characteristics of shipyards, it would be necessary to consider the entire system in light of the six problem areas, taking into account their interaction. Such a task is quite formidable, if not perhaps impossible. Employing a standard, and logical tactic, though, one might consider the problems as being separable and thus obtain at least a first cut on the job shop system.

This thesis will be devoted to the study of just one of the problem areas previously mentioned, that of scheduling and control of the equipment and throughput in a job shop. The scheduling of job shop processes is generally regarded as one of the most complex and difficult of industrial scheduling problems. The complexity stems from the fact that each order may require a processing sequence that is different from the others so that both the scheduling and control functions of the job shop must be capable of dealing with tremendous variations of sequences, processing requirements, number of operations, etc. In order to avoid some of the complexity associated with job shops, this

thesis will focus on the analysis of existing facilities, wherein the location of equipment and the sequence of the throughput are somewhat more restricted than they would normally be in the design of a new facility.

To make the analysis of the job shop problem somewhat more realistic and tractable, as well as to demonstrate the methodology developed, the steel processing facilities of a shipyard were selected as the basis for the analysis and resulting mathematical model. It appears reasonable to select the steel processing facilities, for in doing so, one is able to consider a fairly "typical" production process. The "typical" nature of the steel processing facilities demonstrates the following characteristics: 1) the throughput is variable in that more than one type of throughput flows through the system; 2) the service times of some processes are independent of throughput, whereas others are a function of the throughput; 3) the service time distributions range from deterministic for some processes to completely random (with the associated mean and variance) for other processes; 4) limited storage may exist between facilities; 5) finite amounts of time are required to transit distances between facilities; 6) sequencing alternatives may exist at some facilities, whereas fixed sequences are required elsewhere; and 7) the throughput and processes may be able to be controlled at various locations.

The steel processing facilities will be understood to encompass the steel storage areas within the shipyard and all the processing equipment up to and including those just prior to the subassembly and assembly areas. The most general breakdown of the steel processing facilities is indicated in Figure 1 as a group of activities in series.



Steel Processing Facilities

Figure 1

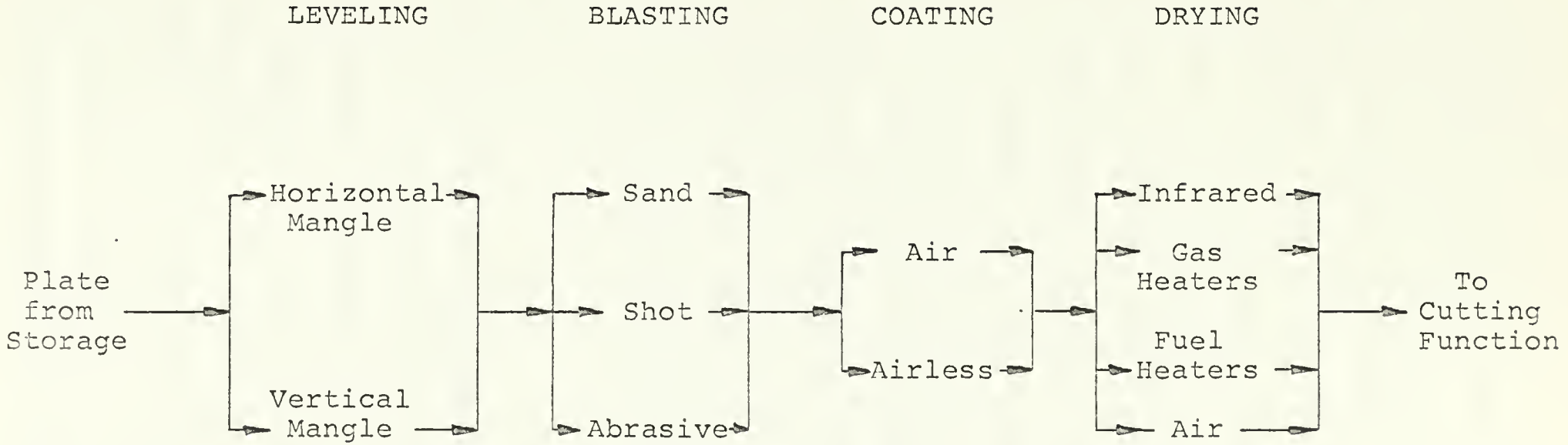
Upon closer inspection, however, one finds that the three major processes are actually comprised of groups of activities which require the use of a variety of resources. This is indicated in Table 1. The individual activities can further be considered to be the elements of subnetworks which describe the processes. An example of the network structure of the preprocessing function is indicated in Figure 2. (Of course, not every element of this process would be present in an actual shipyard, but in the design of a new facility, one would certainly be compelled to consider all the activities indicated.)

The flow in these subnetworks is rather complex in that the throughput may or may not pass through all the branches of the network. The complex flow is only a small

Preprocessing		Cutting		Forming	
Leveling	Horizontal Mangle	Profile Marking	Manual	Roll	Roll Press
	Vertical Mangle		Automatic		Ring Press
Brushing	Wire Brush	Profile Cutting/ Straight Stripping/ Edge Cutting	Gas		Forming Press
	Scraper		Plasma	Bending	Ring Press
	Water Jet		Mechanical		Forming Press
Blasting	Shot		Laser		Frame Bender
	Sand			3-Dimensional Forming and Bending	Forming Press
	Abrasive				Frame Bender
Coating	Air Paint				
	Airless Paint				
Drying	Infrared				
	Gas Heaters				
	Fuel Heaters				
	Air				

Elements of Steel Processing Process

Table 1



Subnetwork of Steel Preprocessing Activities

Figure 2

part of the overall problem, however, as the real life parameters that describe the system do not conform to the most convenient mathematical forms, such as the exponential. For example, some activities might be described as having a uniform distribution of service time, such as the coating activity, whereas other activities have service times that are a function of the throughput itself, such as may be the case with the leveler. The existence of conveyors of finite lengths and speeds results in an added difficulty which is normally neglected in most queuing models. The attempt to describe the throughput rate of such a combination of activities rapidly becomes a formidable task indeed and to further consider the control of the throughput of such a system rapidly leads one to the brink of knowledge with respect to the analysis of such a system.

B. Objective

In order to overcome the apparent stigma caused by the existence of job shop problems, and in hope of offering some viable and acceptable alternative to the decision-maker who is perhaps relying too heavily on intuition, the objective of this thesis will be to reveal or develop a methodology for the analysis of variable throughput production processes.

A great deal of importance will be placed upon a methodology that is acceptable to the decision-maker. For if

it should come to pass that the methodology can accurately describe a job shop system and even indicate optimal control procedures, the methodology will be virtually worthless if the decision-maker refuses to use it. With the goal of acceptance by management in mind, the following set of criteria might be considered the minimum to be met by any methodology that is to have even a chance of survival in the real world, and hence application to job shop problems.

The methodology must be understandable by management, and rely on an orderly and logical pursuit of the problem at hand. The methodology must be capable of realistically describing the processes and throughputs of the job shop, without resorting to simplifying assumptions that reduce the problem to one of little practical value. In particular, the actual time distributions, finite storage capacities, throughputs, etc., must be accurately described. A premium will be placed on the approximate answer to a real world problem as compared to the exact answer to a hypothetical problem. Finally, the methodology must be capable of being applied to production processes in such a way that the time and effort invested in the analysis is consistent with the expected benefits of such a study.

In the pursuit of the stated objective, a methodology for the analysis of variable throughput production processes,

the following format is followed:

1) A literature search indicating the methods of analysis employed to date for the study of the scheduling and control aspects of the job shop problem.

2) An evaluation of the methods of analysis employed in 1) above.

3) Proposal and development of a methodology that conforms to the desired criteria for such a methodology.

4) Conclusions and recommendations regarding the methodology.

II. SURVEY OF LITERATURE

It appears that to date job shop research has centered about three basic techniques: linear programming, dynamic programming, and queueing theory. Linear and dynamic programming have been applied mainly to the problems of pure scheduling (or sequencing). When the job shop is viewed as a network of queues, however, more flexibility arises in the analysis of scheduling problems, yielding information that is potentially valuable from a control point of view. Very little appears to have been done in the area of control theory with respect to the analysis of variable throughput production processes. It appears, as will be indicated, that all the above techniques have been employed under restrictions that are rather severe when one considers the realities of typical shipyard processes.

A. Linear Programming

The general job-shop problem can be stated as follows: Suppose there are m machines of different types, m_1, $m_2 \ldots m_m$ and n jobs j_1, $j_2 \ldots j_n$, each job requiring the use of a subset of the machines in some ordered sequence. A sequence may include return to some machine or machines in the sequence. In addition, it is also possible to have several alternative sequences for the same job. The problem generally considered is one in which it is desired to determine

how to schedule the jobs on the machines so as to finish all the jobs in the smallest lapsed time. Other optimality criteria, however, may be used instead.

One begins by assuming that a period of time can be chosen sufficiently small such that all processing times on the various machines may be represented as an integer-multiple of this "small" time period. Next, a network representation is constructed to represent the different ways a particular job may be processed on the machines, one network for each different job. A route through a given network represents a feasible solution in the completion of a particular job. Each route would, of course, represent a column in the simple tableau, but rather than explicitly generating all the possible routes, advantage is taken of shortest route algorithms. Dantzig [14] demonstrated that the methods first developed by Ford and Fulkerson for multi-commodity transshipment problems were appropriate for the solution of job-shop problems. Here machine sequencing with delays is represented by a network, thus permitting a compact representation of many activities. Further, the shortest route algorithms permit the solution of such problems with relatively little computational effort, thus this approach is very efficient.

The linear programming model employed may be illustrated as follows. Suppose there are two jobs, j_1 and j_2 with the

required sequences,

$$m_1 \text{ then } m_2 \text{ for } j_1$$

$$m_2 \text{ then } m_3 \text{ for } j_2$$

where only one unit of time is required on each machine for each job. There are six possible ways to sequence each of the jobs in the four periods or less required to complete both jobs. The tableau for this problem is illustrated in Figure 3.

The first six columns represent the possible ways to sequence the first job, and the remaining columns represent the possible ways to sequence the second job. If in the solution to the linear program, $x_j = 1$ or 0, it is interpreted that the particular sequence is or is not chosen. Assuming an integer solution, the first pair of constraints indicate that only one sequence will be chosen. The initial objective is to minimize machine use in the fourth period, then the third period and so on. This procedure corresponds to the shortest path route.

The weakness with this particular linear programming model is that fractional basic variables may be obtained, which might call for a half of a job being performed on one machine, and a half of a job being performed on some other machine. If such a subdivision is possible, then the solution is optimal. However, if such a solution is not possible, one is then forced to employ the techniques of

i		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	
1	j_1:	1	1	1	1	1	1							=1
2	j_2:							1	1	1	1	1	1	=1
3	m_1 1	1	1	1										≤1
4	2				1	1								≤1
5	3						1							≤1
6	m_2 1							1	1	1				≤1
7	2	1									1	1		≤1
8	3		1		1								1	≤1
9	4			1		1	1							≤1
10	m_3 2							1						≤1
11	3								1		1			≤1
12	4									1		1	1	≤1
Min	(z)													

Figure 3. Linear Programming Tableau for m = 3, n = 2 Job Shop

integer programming, which provide the possibility of computational nightmares. It has been recommended though [14] that in practical problems, where fractional solutions cannot be admitted, that a rounding procedure be used. If the time completion is not much greater than for the fractional solution, it can be accepted in place of the true optimal which lies between the two completion times. It is not clear, however, what policy should be followed in cases of large numbers of jobs.

B. Dynamic Programming

The use of dynamic programming in the analysis of job-shop problems is similar to the application of linear programming in that it is restricted in use to the analysis of the pure scheduling problem. The problem to be solved is that of scheduling n jobs each requiring a number of sequential operations on m machines in a minimum amount of time. As a result of the inherent computational difficulty associated with problems of this sort, practical means exist only for handling situations in which the number of machines is quite small, i.e., two or sometimes three.

In his application of dynamic programming to the job-shop problem, Devanney demonstrated [6] that a solution to the n job, m machine problem is computationally feasible for cases up to about m + n equal to 10. A significant restriction is placed upon the model: that is, the subsets of

schedules to be considered are those in which only the sequence in which only the sequence in which each job is assigned to the first machine is a variable. The downstream queues are served on a first-come, first-served basis, and no interchange of positions in downstream queues is allowed.

Since a schedule of this type is determined by the order in which each job is assigned to its first machine, it is often called a permutation schedule.

The stage variable for the dynamic program is called k, the number of jobs already scheduled. The state of the system for a permutation schedule is described by two vectors, $\bar{C}$ and $\bar{T}$. $\bar{C}$ represents the jobs that remain to be scheduled at stage k, and is normally described as a vector of the indices of the jobs not yet scheduled. The number of values that $\bar{C}$ can take on is 1 at k=0 and at k=n, and a maximum of $\binom{n}{n/2}$ occur at k = n/2. The vector $\bar{T}$ is represented by a set of times t_j j=1,...m where t_j is the time that the j^{th} machine will become available, given the decisions already made. $W_k(\bar{C},\bar{T})$ is next defined to be the minimum time to completion of the last job given that the jobs not in $\bar{C}$ have already been scheduled, and that the machines are committed until the times indicated by $\bar{T}$.

For k=n, no more jobs remain to be scheduled. $\bar{C}$ is empty, and the minimum time to completion of the jobs is the

maximum of the $t_j \in \bar{T}$. Hence, the boundary condition at k=n is:

$$W_n(\Phi,\bar{T}) = \max (t_1 \ldots t_m) \text{ for all possible } \bar{T}$$

For k=0 to n-1 the recursion relationship is given by the following two vectors:

$$(1) \quad \bar{J}_i = \left\{ j_{i1} \cdots j_{i\ell} \cdots j_{iL_i} \right\}$$

where $\ell = 1, 2, \ldots L_i$, and $j_{i\ell}$ is the index of the machine required for the ℓ^{th} operation of job i. In other words, $\{j_{i\ell}\}$ describes the sequence of operations of job i. The fact that $\ell_i \leq m$ indicates that all jobs need not go through all the machines.

$$(2) \quad P_{ij} = \ldots . p_{ij} \ldots .$$

where p_{ij} is the processing time for the i^{th} job on the j^{th} machine. Given this information the following recursion equation is developed:

$$W_k(\bar{C},\bar{T}) = \min_{i \in C} \left\{ \begin{array}{c} W_{k+1} \ (C-\{i\}) \ , \ \bar{T}_i' \\ \\ 0 \leq k \leq n-1 \end{array} \right\}$$

where $\bar{T}_i'$ is related to $\bar{T}$ as demonstrated in the following three-machine example.

In the three-machine case there are six possible sequences of operation for job i (six possible values of the vector $\bar{J} = \{j_{i1}, j_{i2}, j_{i3}\}$). Suppose now that $\bar{J}_i = \{3,1,2\}$ then if job i is scheduled next, given the present state is

$(\bar{C}, \bar{t}_1\ \bar{t}_2\ \bar{t}_3)$, the third machine will be committed until $\bar{T}_3' = t_3 + P_{i3}$ at which time the job will become available for the first machine. Hence $\bar{T}_1' = \max (t_3 + P_{i1}, t_1 + P_{i1})$ the latter case holding if the first machine is still occupied when the job is ready for this operation. Similarly, $\bar{T}_2' = \max (t_1' + P_{i2}, t_2 + P_{i2})$.

The procedure then is to calculate W_n as indicated above by the recursion equation until $k = 0$. The optimal schedule can be extracted from the optimal table in the conventional dynamic programming manner.

The computational feasibility of any dynamic program depends critically on the size of the state space. As mentioned previously, $\bar{C}$ takes on a maximum of $\binom{n}{n/2}$ value at $k = n/2$. Values of this binomial coefficient range between 20 for $n = 6$ to 252 for $n = 10$ and 924 for $n = 12$. Thus, if the number of values which the individual t_j can take on at stage k to, say ten, then a $n = 10$, $m = 3$ size problem could be done on a machine containing 500k words of primary memory [16].

If one considers the subproblem of n/m job shop termed the flow-shop problem, that is, the case where all jobs follow the same sequence, a simplification results in the problem formulation. The state space no longer requires the variables $\bar{T}_1$ since the order of the jobs to be performed on the remaining jobs, $\bar{C}$, to be scheduled is fixed. As a

result, the dimensionality of the problem is reduced by one, and since the n = 10, m = 3 problem could be solved before, the n = 10, m = 4 flow-shop problem can now be solved. Further, the state spaces on the remaining T_j's become triangular, with an attendant decrease in memory requirements. As a result of the savings of a great deal of memory space, problems on the order of n = 16 or 18 and m = 3 can be solved. Thus, it is the opinion of some [16] that the dynamic programming approach might be a worthy competitor to the branch and bound algorithm for the m = 3 flow-shop problem.

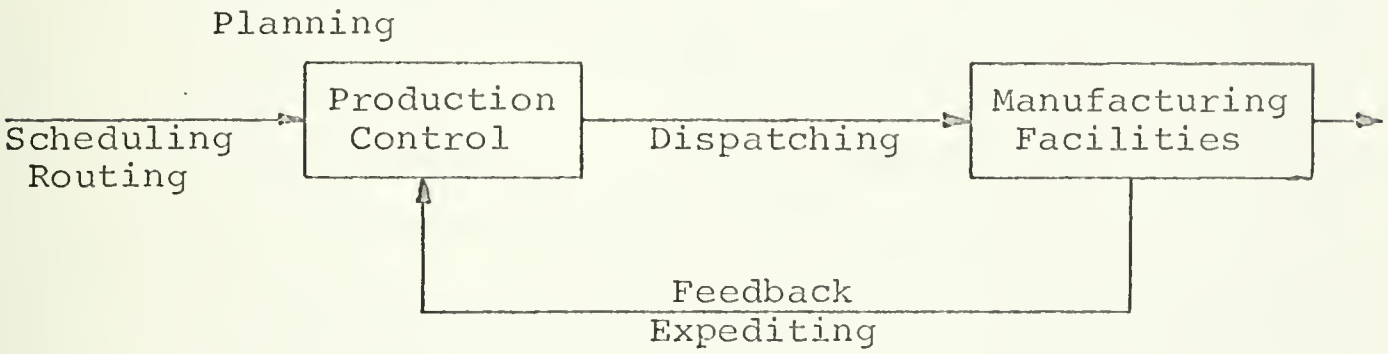
It is noted that various algorithms exist that have been based on dynamic programming formulations in order to deal with some rather simple n/m=3 flow-shop problems, but they are of little value in the study of large job-shop problems.

C. Control Theory

In the process of searching for an effective method of analysis for variable throughput production processes, one tends to become rather optimistic upon finding the following statement by Greene in his book Production Control Systems and Decisions [22]:

> "One of the most important and useful analogies for the industrial system designer is the servo system. The servo system is a control system with two features

amplification and feedback ... Typical of the production system to be controlled is one which possesses the production control functions of routing, scheduling, dispatching and expediting interacting with the manufacturing facilities. (Such a system is illustrated in Figure 4.) ... [Through the application of control theory] one is able to achieve the objective of production control; the coordination of the production facilities to produce a product on schedule at an optimum cost."



Production Control Network

Figure 4

After reading Greene's comments, and the remainder of his book which tends to give the impression that the application of control theory is the panacea for all production control problems ranging from public relations to computer applications, one is obliged to investigate, in greater detail, the field of control theory. Initially one is encouraged, as the preliminary presentations regarding control systems seem to indicate a capability of handling almost any system for which one might be able to describe the input and transfer functions. For example, Newcomb [39] begins the investigation of a generalized system by use of the following example:

A system may be defined by:

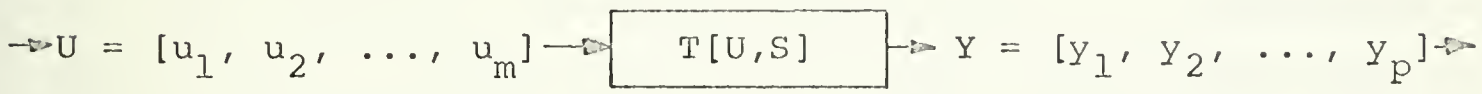
U a vector that describes the inputs of a system comprised of elements $u_1, u_2, \ldots, u_m$

S A vector that is comprised of the parameters $(s_1, s_2, \ldots, s)$ necessary to describe the initial conditions of the system.

T a transfer function which transforms the inputs, based on the initial conditions of the system, into a set of outputs

Y a vector that describes the output of a system comprised of elements $y_1, y_2, \ldots, y_p$

Such a system is indicated by Figure 5.



Generalized Control System

Figure 5

In pursuit of the problem one finds what might be the last accommodating feature of control systems, the description of time as a continuous variable. Unfortunately, though, in the application of control theory to even the most basic of problems it is imperative that simplifying assumptions be made in order to keep the problem computationally feasible. Further, when dealing with complex

production problems, such as those characteristic of shipyards, the simplifying assumptions result in a gross departure from the actual system under study. Thus the value of such methods of analysis for the job-shop problem is somewhat questionable.

Of course control theory can be applied to many complex real world problems, such as demonstrated by Beusch [4] in the study of the dynamic behavior and control of communications networks, but the reason for success in such application is that the necessary simplifications are not inconsistent with the system under study.

D. Queueing Models

1. Queueing Theory

The recognition that the job shop could be represented as a system or network of queues was most significant, for it touched off a great deal of research in the analysis of job shop problems. Unfortunately, though, a clear-cut and purely analytical method could not be adopted using what might be called classical queueing theory. It appears that the restrictions placed upon even the most general of the theoretical queueing models, such as those of Satty [46] and Prabhu [40], result in descriptions of systems that are rather far removed from the realities that exist in actual job shop situations. When one desires to venture beyond Poisson arrivals and service times, to situations that more

accurately describe these phenomena, such as processes with time distributions of a given mean and variance, or if one desires to investigate queues that cannot be modeled merely by use of the Markov models, the theoretical approach becomes totally unmanageable.

Through the methods of computer simulation, though, it becomes possible to conduct large scale evaluations of queueing models, taking into account factors that previously had to be assumed away. The early research pointed up the queue discipline as being an important variable which could be manipulated in sequencing models [5]; thus research beyond the first-come, first-served queue disciplines was initiated. Additionally, the constraints imposed on the job shops as a result of the system being machine limited or labor limited were incorporated into the queueing models.

2. Simulation of Queueing Models

The analysis of job shop systems is normally undertaken subject to a constraint either on machines or on labor. When the constraint is placed upon the total number of machines, it is assumed that labor is always available. Conversely, when labor is considered the critical resource, it is assumed that there is always an adequate supply of machines; hence, the labor-limited system.

It is interesting to note that the first comprehensive simulations conducted in the analysis of job shop systems

were conducted, taking into account the resources of both machines and labor, but that no assumptions were placed on arrival rates. Of the subsequent studies, however, it appears that most of the work by Baker and Dzielinski [2] in 1960, Conway and Maxwell [12] in 1962, Nannot [5] in 1963, and Carroll [7] in 1965 has been in the context of machine-limited shops involving assumptions of Poisson arrival rates and exponential service times. Dual resources of labor and machines appeared again with studies by Allen [1] and Nelson [36]. Labor-limited systems have also received consideration by Rowe [44], Allen [1], and Le Grande [29]. Finally, Harris [23] conducted an extensive investigation of an actual plant in order to examine the assumptions involved in the majority of the queueing models to that date.

It would be impossible to consider here all the simulation studies that have taken place. Rather, the approach will be to consider in a general way the results of the machine-limited and labor-limited systems, in addition to noting the major conclusions of the Harris study regarding queueing model assumptions.

The study of the machine-limited job shop revolved about the examination of various queueing disciplines in order to achieve some objective. The objective was frequently taken as minimizing the time required to finish all jobs, minimizing the average delay of completion beyond a

certain time, or minimizing the maximum delay in job completion. There are basically two methods of classifying the dispatching rules (queueing disciplines). One is based on an horizon, and is termed a "local rule" as it determines priority entirely on the basis of the characteristics of the order in question, for example, its processing time or due date. The second method of classification is a static rule, as opposed to the dynamic rule already discussed. In static rules, the relative priorities remain the same once assigned. For example, first-come, first-served is static in that the priority of a job will always remain the same upon arrival at any queue. Its relative position with respect to its completion time, or the slack remaining in its schedule does not affect its priority.

Most studies of the machine-limited system involved two or more job shop structures and the testing of various dispatching rules. Typical job shop structures are identified by the number of work centers, the degree of loading (high, medium, or low) and a statement indicating whether the shop is a pure job shop, quasi-flow shop, or a pure flow shop. The more commonly considered dispatching rules are indicated in Table 2.

Typical explicit assumptions for machine-limited systems might include: the arrivals at a machine are Poisson-distributed and the service times are negative

exponentials; the number of queues for each machine is limited; lot splitting is not allowed; transportation times between machine centers is zero; no subcontracting or overtime is allowed; machine breakdowns, scrap, and other interruptions are not allowed; setup time is considered to be part of the processing time for each operation, etc.

Table 2

Queueing Disciplines

(1)	FCFS	- First come, first served
(2)	SOT	- Shortest operating time
(3)	SS	- Static slack, that is, due date less time of arrival at machine center
(4)	SS/PT	- Static slack/remaining processing time
(5)	SS/RO	- Static slack/remaining number of operations
(6)	FISFS	- Due date system; first in system, first served
(7)	LCFS	- Last come, first served
(8)	DS	- Dynamic slack (time remaining to due date less remaining expected flow time)
(9)	(DS/PT)	- Dynamic slack/remaining processing time
(10)	(DS/RO)	- Dynamic slack/remaining operations

A typical result of such a study would be the mean flow times and associated standard deviations of the decision

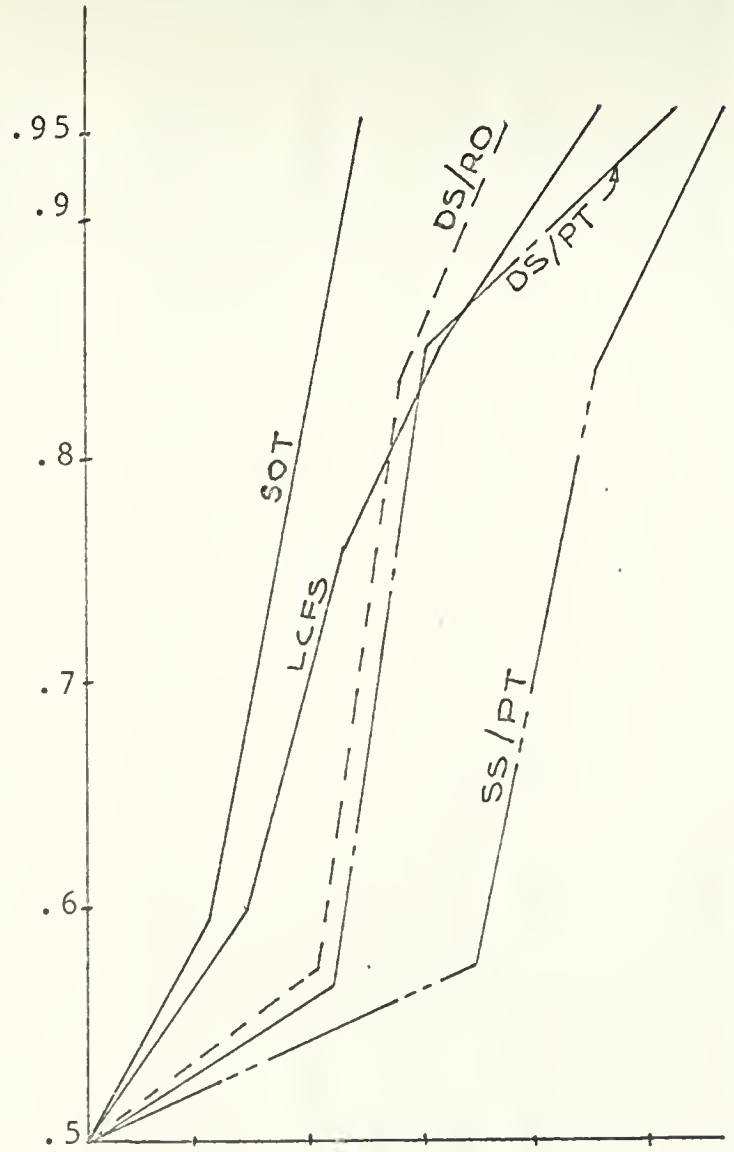


Figure 6

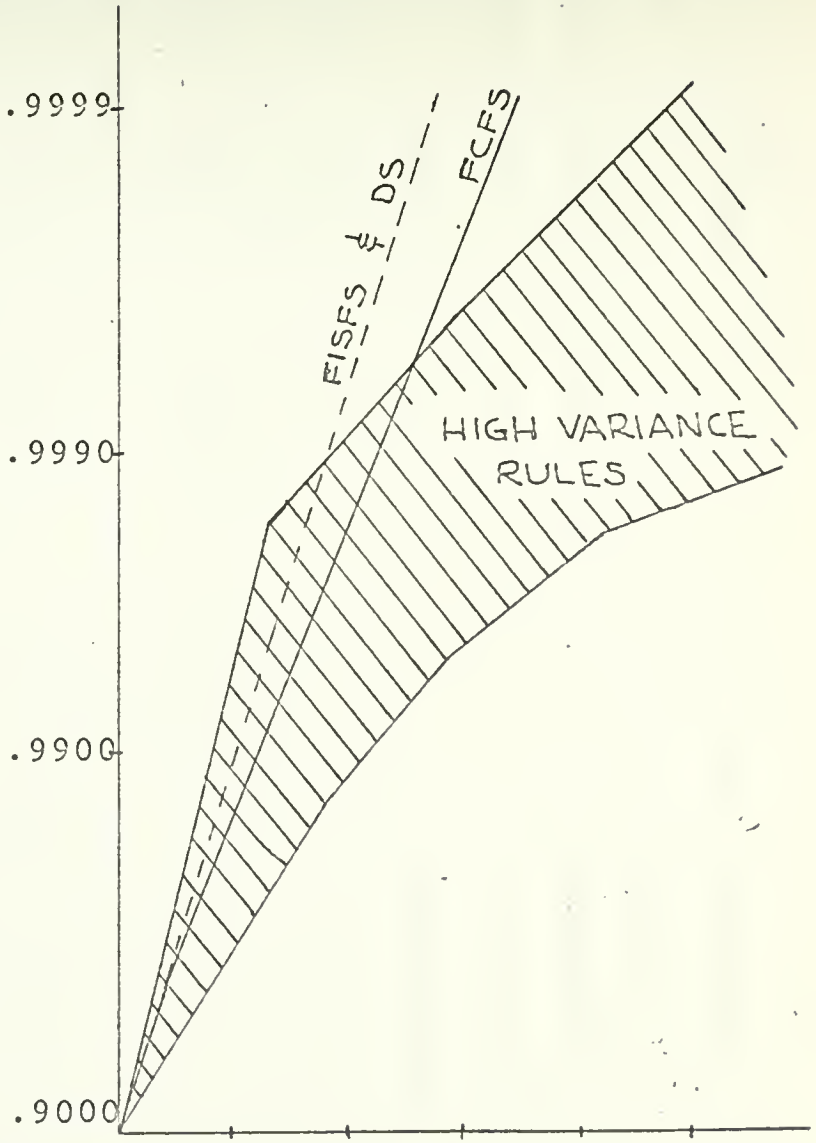


Figure 7

Results of the Simulation of Job Shops Conducted by Nanot

rules for the various shop structures. Figures 6 and 7 show the results of the simulation conducted by Nannot [5]. It can be stated that these results are generally those achieved by other investigators in that it was found that the shortest operating time, SOT, rule consistently has the lowest mean flow times, and that the standard deviations of the FCFS and FISFS rules are in general lower than for other rules. In addition, systems are often investigated by the analysis of the variance of the mean flow time, which then becomes the measure of effectiveness for the system. Finally, the results of the various investigators showed rather dramatically that the differences due to the use of different priority dispatching rules was highly significant, but that the differences due to system configuration, shop size and differences in routing, for example, were very much less significant.

Turning our attention now to the labor-limited job shop systems, one finds that in many instances this is the more appropriate model of "reality". It is quite usual that not all machines are manned simultaneously, labor being used as a flexible resource assigned to operate different machines at different times, depending upon the needs of different orders.

In addition to the priority dispatch rules, a labor-limited system raises other questions regarding the

functioning of the job shop. For example: "What are effective assignment procedures and how do these procedures interact with sequencing rules? What are the effects of various degrees of centralized control?"

As a result of the additional questions posed by the labor-limited systems, it is necessary to consider a larger set of design and control parameters than was necessary for the machine-limited system. The design parameters now include, in addition to the number of machines and machine centers in the system, items such as the number of laborers in the work force and the relative efficiency of a laborer on a given machine. The control parameters now include, in addition to the queue discipline, the degree of centralized labor assignment control exercised at a particular machine center, and the machine center selection procedure used in central control.

Again, the majority of the labor-limited systems have assumptions regarding the Poisson arrival and exponential service rates, with variations in systems from pure job shops to pure flow shops [1], [7], [12], [29].

The results of the labor-limited systems are, in general, consistent with the results one might expect when considering the machine center selection procedure, and changes in the shop size seem to have little effect on the performance of the systems with respect to the mean and

variance of the flow time. Changes in queue discipline again showed substantial effects. (It is noted, though, that Nelson [37] obtained results for the mean and variance of flow that were quite opposite to those generally obtained by the machine-limited systems.) Finally, it was discovered that changes in the labor force resulted in the most significant effect on the mean and variance of the flow. For the most effective queue disciplines, that is, FISFS, FCFS, and SOT, it was found that increasing the manning level by 25% resulted in large increases in both the mean and variance of the flow [37].

In the case of all the machine-limited studies considered above, and in most of the labor-limited studies, the job shop was idealized in structure to fit the assumptions of exponential interarrival times and exponential service rates. That is, each of the models was assumed to fit the Erlang family of mathematical models. Harris [23] performed an empirical study aimed at three specific questions: 1) How well do the Erlang structural assumptions fit the work center? 2) How well do the Erlang model parameters fit the work center? And 3) How well does the Erlang model predict the observed behavior?

Harris found that much of the general structure paralleled that of a network of queues, but that some of the implications of actual practice did not fit assumptions.

For example, it was difficult to translate service time to a service rate because of the interruption of service and the rapidly fluctuating capacity of the machine center. Queues did, in fact, form in front of most of the machine centers, but the reasons for queueing were more complex than just waiting for service. That is, some jobs may have been waiting for the service to "open" for business, or to satisfy other management requirements. Harris' conclusion was that the general Erlang model did not fit the work center very well.

Harris also performed an analysis of the observations classed by machine centers to determine if the arrival and service rates corresponded to a Poisson process. Based on an analysis of mean arrival rates, standard deviations of arrival rates and chi-squared fit tests, the conclusions reached were: a) the observed distribution of arrivals shows wide variance in the arrival rate, variability, and general shape of the distribution; b) the Poisson type distribution is not sufficient to explain or fit all the observed distributions. With regard to the negative exponential distribution of service times, Harris came to similar conclusions.

Harris also noted that the observed waiting times and queue lengths are longer than predicted by the Erlang model. The observed standard deviations are different from those

predicted by the Erlang model, the queue length variation being less and the variation in waiting time being longer. His final observation was that parametric changes in the Erlang model do not appear to be able to generate a single distribution which can predict all the distributions.

E. Evaluation of Previous Methods

Although each of the techniques discussed thus far appears to have merit when applied to rather restricted cases, none seems to be able to fulfill the minimum criteria for acceptance by the decision-maker confronted with a complex job shop problem. (See "Objective") Perhaps the most significant shortcoming that all these techniques have in common is that they have all made rather severe simplifying assumptions concerning service time distributions and the actual physical properties of the systems under study, i.e., finite buffer storage and transit distances, etc.

Linear programming and dynamic programming are apparently satisfactory for application to the pure scheduling aspects of the job shop problem provided 1) the number of jobs and machines is quite small, 2) the problem can be described adequately by purely deterministic processes, and 3) the queueing discipline is first-come, first-served.

Both these techniques, however, share deficiencies that render them computationally unfeasible for the solution of

the "typical" variable throughput production problem already mentioned. Namely, these techniques cannot cope with the complexity of stochastic arrival and service times, the large problems that involve many processes, nor do they provide information that is adequate for the control of the job shop processes.

Control theory seems to be severely limited for the analysis of variable throughput production processes due to the linearity assumptions that must be placed on the transfer functions (i.e., the description of the process service time distributions) in order to maintain computational feasibility. If it is desired to describe the processes more accurately and adopt non-linear transfer functions, computer simulation techniques are then required. Further, the consideration of physical system constraints such as finite capacity storage areas introduces such a degree of complexity that other methods of analysis are found to be less cumbersome and less time-consuming. It appears also that the application of control theory to the type of problem under consideration might prove less appealing and not as acceptable to a decision-maker if other methods of analysis are available.

Clearly, the only way in which queueing models can be effectively employed in the analysis of job shop problems is by means of computer simulation. The success of the

hueristic approaches to actual production processes has been demonstrated by the investigators already cited, in both the machine-limited and labor-limited situations. Large n/m job shop studies have proven effective; there exists the flexibility to examine numerous queueing disciplines, and a variety of stochastically and deterministically described systems have been studied. Although none of the investigations considered the physical constraints associated with production systems, nor systems that were described by totally stochastic methods, the capability to do so is evident. Finally, it appears that even with some simplifying assumptions, the simulation of job shop processes by the use of queueing models is a method acceptable to and deemed worthy by the decision-makers of the companies concerned.

III. TOWARDS A METHODOLOGY

A. Rationale for Simulation of Queueing Model

After considering the research already conducted it appears that only one of the techniques is capable of handling complex throughput production processes in a computationally feasible manner, while at the same time meeting the criteria of management acceptance. Thus the pursuit of a methodology for the analysis of the production processes of concern here will be based upon the simulation of queueing models.

It is interesting to note that even prior to the advent of the mentioned research Feller [19] considered the technique of the simulation of queueing models to be justified "in light of the capital involved in making facility decisions regarding the operation and purchase of expensive equipments". But Naylor's [35] justification for the use of computer simulation seems to be especially apropos in this case. He recommends simulation if the following three conditions can be met: First that one should be relatively certain that either an exact solution or a satisfactory approximation to the problem can be obtained; second that simulation offers the lowest cost computational procedures to solve the problem; and third that the particular technique under consideration must lend itself to relatively

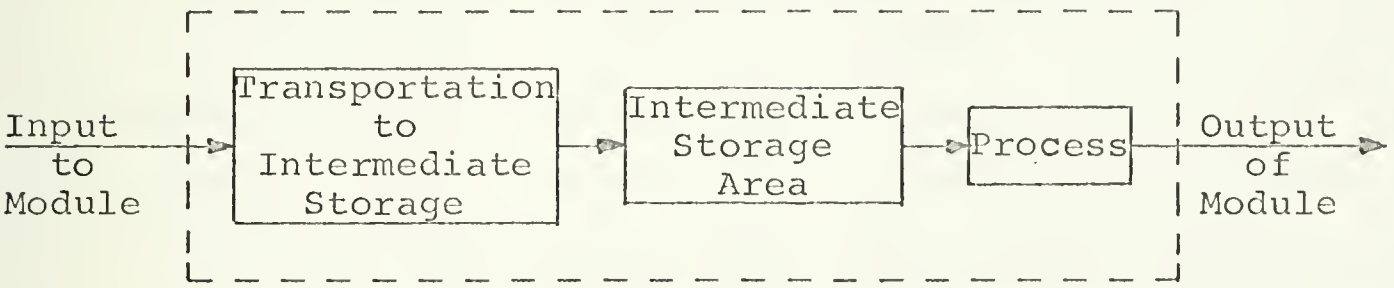
easy interpretation by those who are likely to use the results of the simulation study.

To directly address Naylor's criteria for simulation, which are certainly consistent with those stated in the "Objectives", it is obvious that the first condition is met. The literature has indicated the inability of purely analytical methods to accurately describe complex production processes. Only by the use of simulation can a variety of stochastic processes subject to realistic constraints be adequately described, and thus satisfactory problem solutions expected. The fact that simulation is the only computational procedure capable of dealing with the large queueing models under consideration satisfies the second condition. Finally, the apparent successes associated with computer simulations of production facilities, and the willingness of managers to adopt procedures based on the results of the simulations, indicate that such methods appear to be relatively conducive to interpretation by those personnel likely to use such results.

1. Desirable Characteristics of Model

Before proceeding to the planning stage of the simulation model, it is worthwhile to consider two particular features that would be desirable in a model of this type, and to mention the reasons for not using any of the simulation languages that are already available.

First to be considered is the desirability of developing a model that is composed of modular elements each of which represent a particular process or machine. Thus it might be a relatively straightforward process to model complex systems by placing the modules in their appropriate positions. A schematic of items to be included in a typical module are indicated in Figure 8. Such a building block must have the capability of realistically describing the actual process by including such factors as the length and speed of conveyors, the capacity of buffer storage areas, the arrival time and service time distributions when appropriate, the ability to identify individual elements of the throughput, etc.



Module of a "Typical" Process

Figure 8

Normally, one undertakes the simulation of complex production processes because of some desire to improve their functioning. Once satisfied that the simulator is adequately reproducing reality, one then experiments with the simulation model in the hope of improving some characteristic of

the system. For example, the most common improvement sought is probably an increase in the throughput rate to the maximum possible. Thus, by defining such an objective, a desire to reach some optimality condition is implied.

This then leads to the next desirable feature of the model being considered, the capability to incorporate mathematical optimization techniques. There are two primary benefits of such a capability.

First, significant amounts of computational time might be saved if one were able to employ such techniques in a large simulation, in contrast to merely allowing the simulation model to plod on through time, investigating all the possible outcomes of reality. Perhaps the simulation of trivial or undesirable events can be reduced or even eliminated altogether. At this time, though, there is no unified body of optimization theory for waiting line models [19], and in most applications the optimization technique is applied ad hoc.

Of course, to undertake the task of developing a unified body of optimization theory is beyond the scope of both this thesis and this individual. However, an attempt will be made to incorporate optimization techniques into the model when possible. The use of mathematical programming could be very valuable, for example, in the case where one is considering a series of production processes, among which is

located a small job shop, and thus employ optimization techniques to the sequencing of jobs in the job shop.

The second benefit of using optimization techniques is that at least the objective functions of individual processes are optimized. It is not being implied that the optimization of various associated processes will result in the optimization of some of the processes, for this would not generally be true. What is being implied, though, is that the clever use of optimization techniques where practicable might more readily lead to sub optimal solutions of production systems when compared with the pure simulation of the same systems.

2. General vs. Problem Oriented Computer Languages

When contemplating the use of a digital computer for the simulation of the queueing models necessary to describe shipyard production processes, two options are generally available. One can elect to use one of the existing problem oriented languages, which are identified as simulation languages, or one can elect to make use of a general purpose language such as FORTRAN. The criteria for selection of one method or the other normally is based upon economic considerations such as: 1) availability of computer hardware; 2) availability of programmers knowledgeable in particular computer languages; 3) cost of programming per unit time; and 4) cost of computer time. In this case, though, these

criteria do not result in a clear-cut choice; thus it is necessary to consider in somewhat greater detail the alternatives of the two options.

Among the simulation languages available one finds GPSS [21], SIMSCRIPT [30], GASP [17], SIMULATE [50], CYNAMO [41], etc. The languages possess generalized structures for designing simulation models, a rapid means for converting a simulation model into a computer program, and a flexible way for obtaining useful outputs for analysis. Despite their apparent flexibility and general character, though, they are somewhat restrictive, i.e., DYNAMO and SIMULATE land themselves best to simulations of large scale economic systems, whereas GPSS and SIMSCRIPT are best suited to certain types of scheduling and waiting line problems. Of the latter two languages, GPSS relies upon fixed time increments, and SIMSCRIPT relies upon variable time increment models.

The use of a general computer language offers the programmer maximum flexibility in 1) the designand formulation of the mathematical model of the system being studied, 2) the type and format of output reports generated, and 3) the kinds of simulation experiments performed with the model. The major drawback of using a general language is that one must bear the burden of writing the simulation program. "It is during the writing of the program, when one is enmeshed in the rather uninteresting, but complex details of the

control of the interdependent activities of the sequence, that one is subjected to a very fertile ground for minor errors. Moreover, the mistakes are liable to produce obscure effects and are correspondingly difficult to eradicate." [11]

Considering that an attempt is being made to develop a methodology for the analysis of a particular type of production process, it is desirable to use a computer language that will afford the maximum flexibility by imposing the least number of programming restrictions on the system. In addition, a language compatible with mathematical optimization techniques is desirable. As a result of these two requirements of the computer simulation language, it is mandatory to elect the general computer language approach. The following computer simulation will then be written in FORTRAN IV.

3. Fixed vs. Variable Time Increment Methods

Two general methods are available to move the model of a system through time, the fixed time increment method and the variable time increment method. With the fixed time increment method a clock is simulated by the computer and thus the instant of real time that has been reached in the system is recorded in order to maintain the correct sequence of events. During each increment of time (which may be minutes, hours, days, etc.), the entire system is scanned to

determine whether any events have taken place. With the variable time increment method, the clock time is advanced by the amount of time necessary to cause the occurrence of the next most imminent event. Events can occur at any point in time, be it seconds or days away, because the time of the next most significant event is calculated in advance. Time periods during which no events occur are just skipped over by the simulator.

A good deal of research and controversy has taken place regarding the relative merits and the desirability of using one method or the other in a particular computer simulation. [8], [11]

Despite observations that the fixed increment methods are best suited for systems in which events can be expected to occur in a rather regular manner, it has been shown that the same methods are quite effective in the study of a system whose significant events are not well known. [8] In addition, Conway et al [11] have demonstrated that the efficiency of fixed increment methods increases with the number of status variables.

On the other hand it is generally agreed that the variable time increment methods are most effective for those systems in which the events occur unevenly in time, or in which there are extended periods during which the system remains static [8]. The computational efficiency of variable

increment methods has been shown to increase with the mean length of events [11].

Naylor [35] concludes, however, that the only sure way of determining which method minimizes computer time is to experiment with both methods on the same system application.

After all is said and done, the choice of which time increment method to use might appear to be rather arbitrary, but after careful consideration of the nature of production processes being contemplated, it is apparent that there will be a myriad of status variables. Every position which the throughput can occupy will have a status variable associated with it, so even the computer might shudder at the thought of pre-computing the next imminent event when the events are so minutely defined. It is also difficult to imagine a group of production processes that would be static for long periods of time. Thus, relying on these arguments and anticipating the avoidance of some complex sequencing control problems, the fixed time increment method will be utilized for the simulation of all systems to be considered in this thesis.

IV. FORMULATION OF MODEL

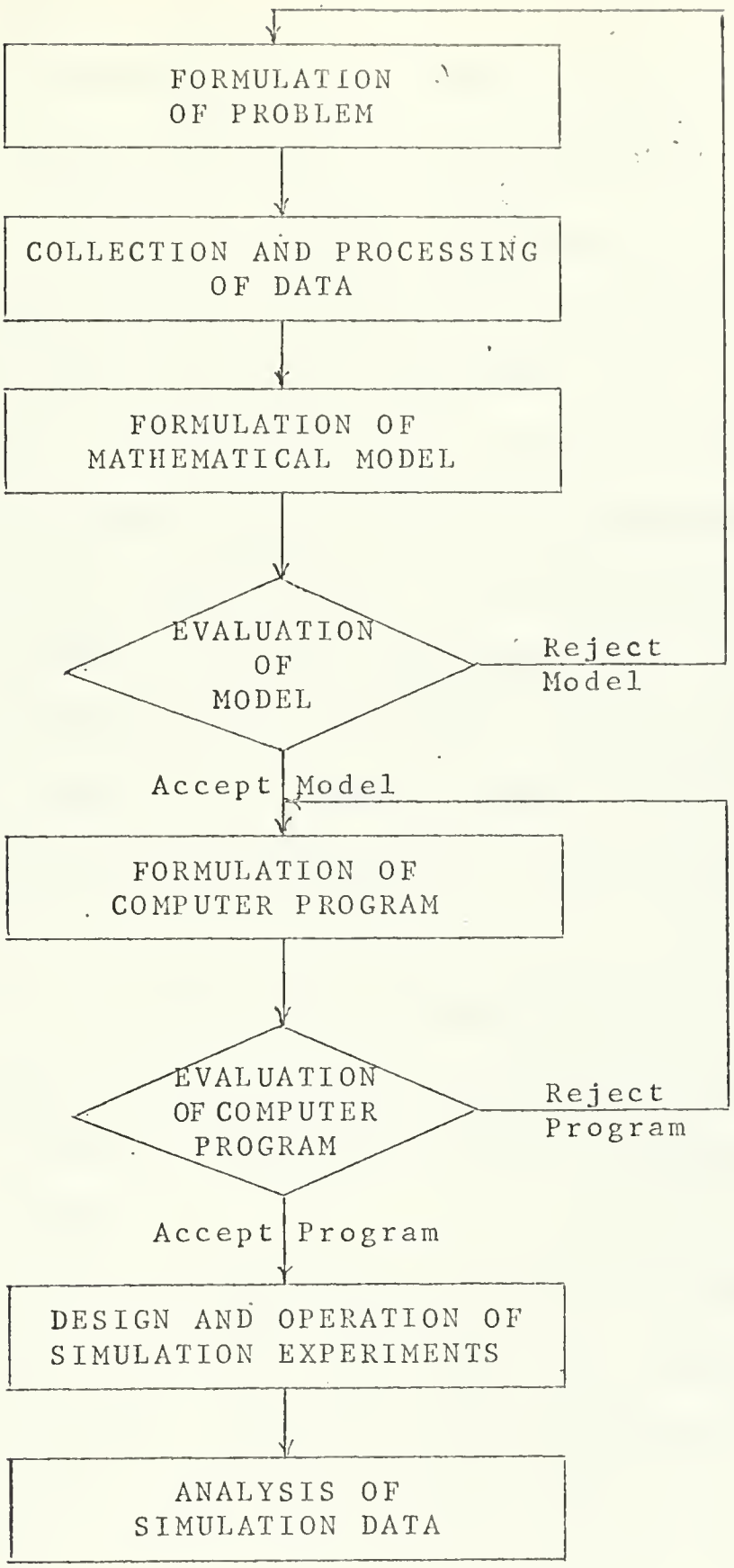
The planning of simulation models is, in general, a rather involved process which requires adherence to a logical method. Such an approach is especially helpful when examining multistage production processes because it provides a kind of checklist of the important factors to be considered. By resorting to such a checklist one can be more confident of a complete and well organized analysis, while at the same time being forewarned of potential pitfalls that may arise when conducting a simulation experiment.

The flow chart in Figure 9 represents the essential elements that will be considered during the planning of the simulation of the variable throughput production processes.

A. Formulation of Problem

Prior to beginning the simulation experiment, the following two important decisions will be made. First, the objectives of the research will be decided upon and second, a set of criteria will be selected in order to be able to evaluate the degree to which the objectives were fulfilled.

The objectives of this simulation are to 1) develop a realistic model that can be effectively used for the analysis of variable throughput production processes, 2) demonstrate the use of the model in the analysis of a production system



Flow Chart for Planning Simulation Experiments

Figure 9

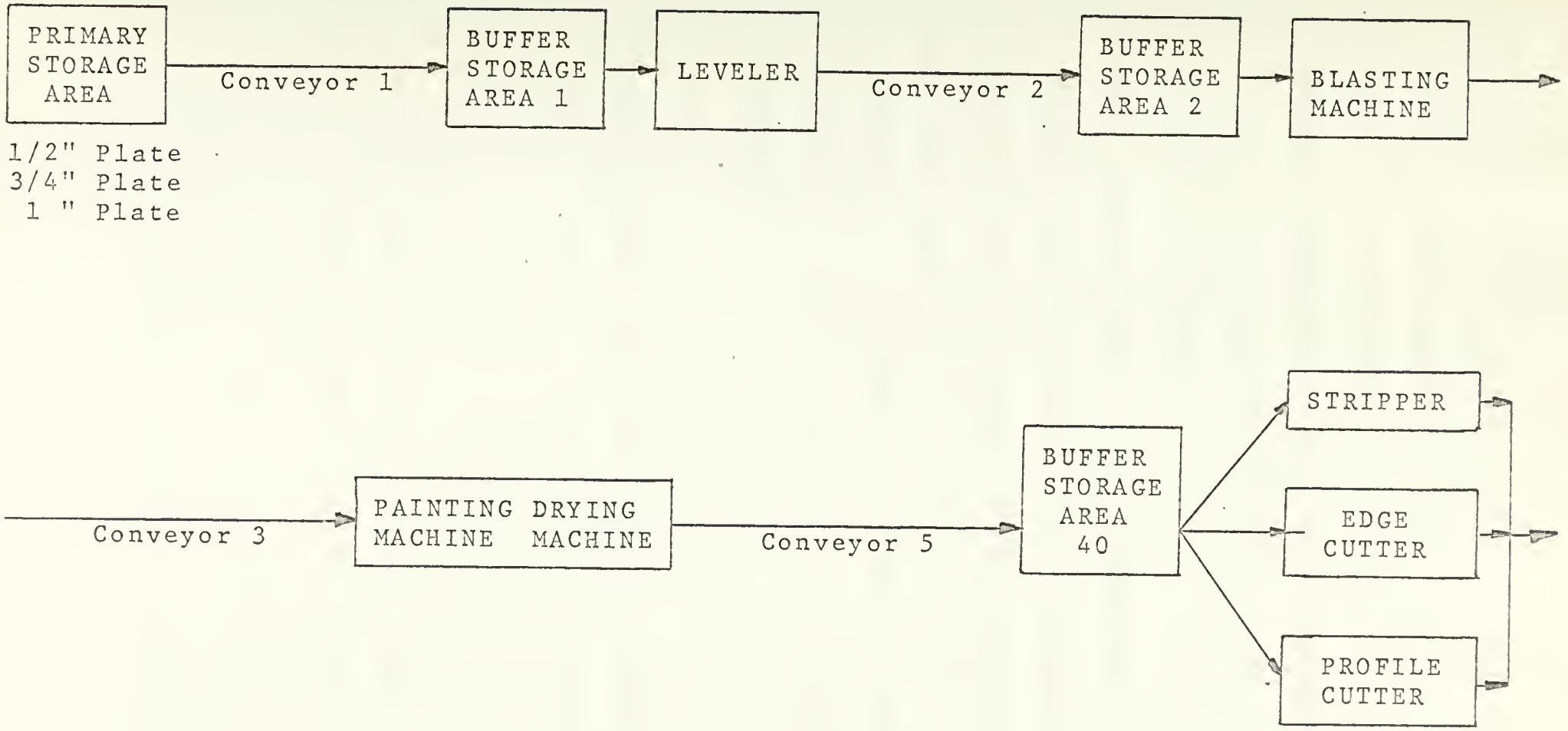
by simulating the conditions of an existing facility and determining the maximum throughput of that facility.

Perhaps the best way to evaluate the success of the model would be to collect the appropriate data from a real world system, and then verify that the model can, in fact, simulate the system. Since this is not possible within the constraints of this thesis, the criteria for determination of model success will depend upon 1) the ability of the model to produce results that are consistent with theory, when theoretical conditions are specified, and 2) a demonstration that the "maximum" throughput rate is indeed greater than the simulated "existing-facility" throughput rate.

Consider now the system under investigation, a portion of an existing steel processing facility which is comprised of the following elements (see Figure 10 for a schematic diagram of this system):

1) The primary steel storage area in which is located a supply of steel sufficient to satisfy the demands of the processing line. Except for their thickness of 1/2", 3/4" and 1", all plates stored there are of the same dimensions.

2) The leveling process is comprised of a conveyor which transports the plates from the primary storage area to the buffer storage area of the leveling machine. The



Schematic Diagram for the Portion of the Steel Processing Facility under Consideration

Figure 10

conveyor has a length equal to three plate lengths and is of the constant speed type. The buffer storage area has a capacity of two plates. The leveling machine can process only one plate at a time; its service time is stochastically described and is a function of plate thickness.

3) The blasting process has a conveyor of two plate lengths which transports the throughput from the leveling machine to a buffer storage area that has a capacity of four plates. The blasting machine is capable of processing one plate at a time; its service rate is stochastically determined but is independent of plate thickness.

4) The coating process is located a distance of three plate lengths from the blasting machine, the plates again utilizing a constant speed conveyor for transportation. No buffer storage area exists. The process of coating plates here includes both painting and drying. The throughput proceeds through these two service areas at a continuous rate (the same speed as the conveyor); therefore the service time is deterministically described. The total capacity of the coating "machine" (painting machine and drying machine) is two units of throughput.

5) The cutting process consists of a constant speed conveyor of length two which transports the plate to a relatively large buffer storage area that has a capacity of forty units. The cutting process machines consist of three machines, a stripping machine, an edge cutting machine, and a profile cutting machine, each of which possesses its own stochastically described service time which is a function of the throughput thickness. The cutting process discharges its throughput immediately upon termination of the throughput service.

B. Collection and Processing of Data

The following characteristics, which are assumed for the present analysis, are those which one might expect to obtain after collecting and processing the real world data of the steel processing system.

Steel plates, 20 feet long and 8 feet wide, arrive at the first conveyor in a completely random manner. It is observed that the shortest interarrival time is 2 units and the longest is 10 units. The uniform distribution (see Appendix B) is selected, with a = 2 and b = 10. Furthermore it is observed that the different plates arrive according to the probability mass function (PMF) indicated in Figure 11.

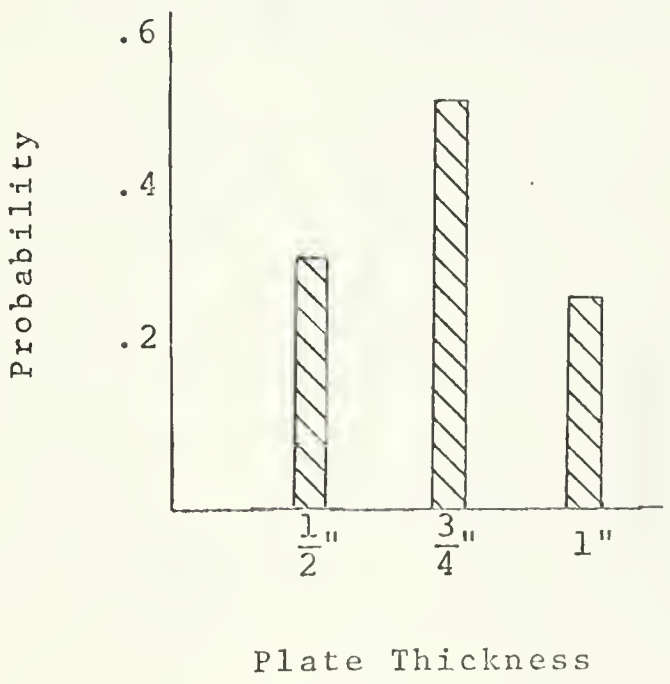
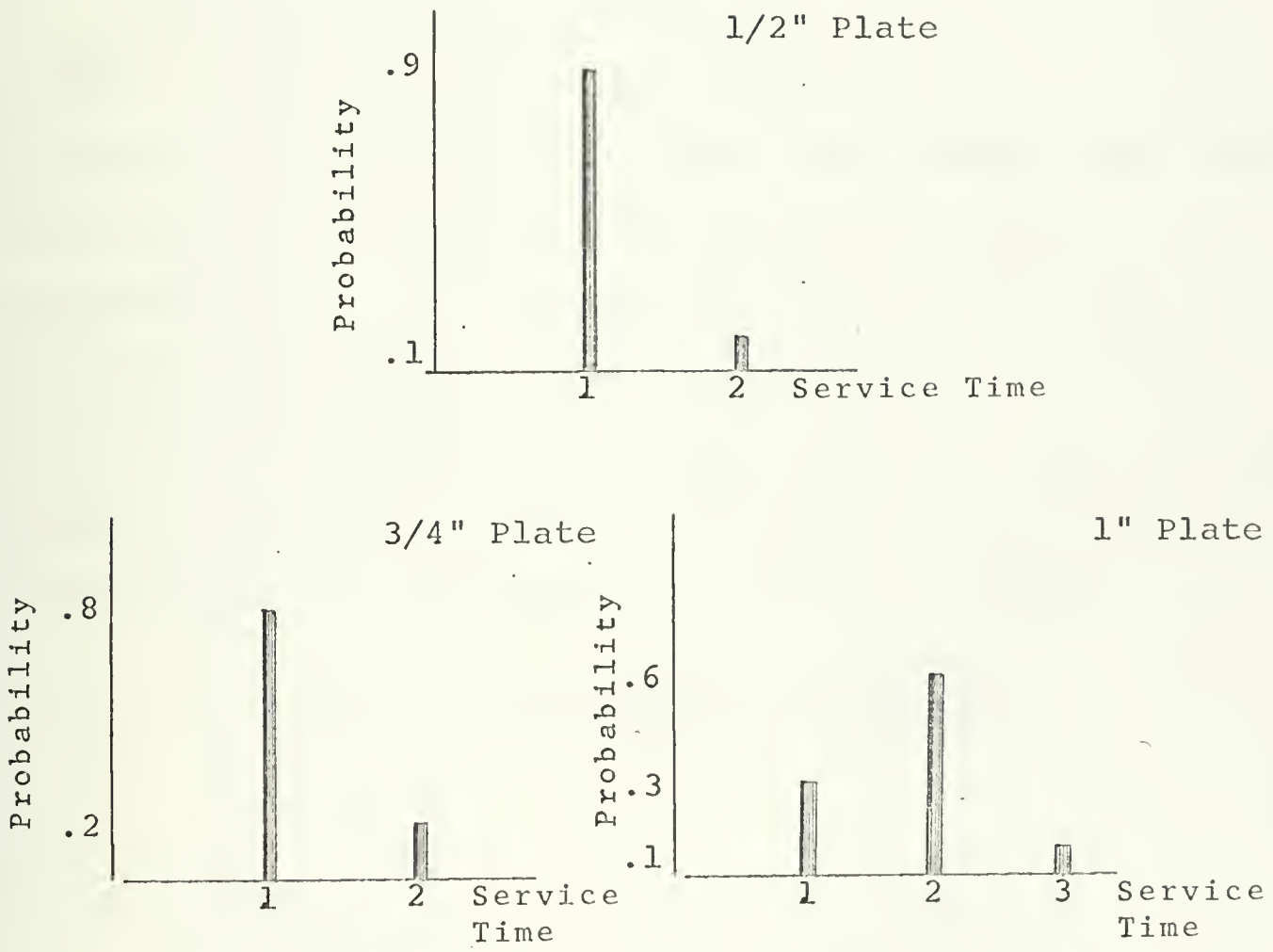


Figure 11
Distribution of Plate Sizes

All conveyor speeds are identical and constant. For convenience, all times in this analysis have been normalized to the time required for a plate to move one plate length on a conveyor.

The leveler is observed to have service times that are a function of the throughput size and are described by the PMFs of Figure 12.



PMFs for Leveler Service Time

Figure 12

The blasting machine service time is independent of throughput type and is observed to be described by a normally distributed variate with a mean value of 1.5 and a standard deviation of .15.

The machines of the coating process, i.e., the painting machine and the drying machine, have deterministic service times of one unit each.

Within the cutting area the flow of plates is noticed to be quite complex due to the number of machines, the percentage of each type of plate that flows through each, and the service time dependence of the various machines on the throughput size. As a result, for a first cut on the problem the data gathered on the cutting shop is restricted to the service time distributions for each plate type. It is observed that the service time distributions are normally distributed with the parameters indicated in Table 3.

Plate Size	Mean Service Time	Standard Deviation
1/2"	9.0	2.0
3/4"	6.0	1.4
1 "	8.0	.5

Service Time Parameters for Cutting Process

Table 3

C. Formulation of Mathematical Model

The formulation of the mathematical model involves relatively few assumptions and is rather straightforward. The assumptions that are made are reasonable in that they do not result in oversimplification of the model. Furthermore, if the assumptions were not made, a significant increase in computational effort would be required due primarily to the combinatorial nature of the problem. It would then be difficult to justify obtaining the results of such a precise model in light of the large expenditure of resources that are required.

It is assumed for this model that the primary plate storage is large enough to supply plates indefinitely, in accordance with the specified arrival times and plate distributions. The last stage of the system, here the cutting process, is allowed to discharge its throughput as soon as processing is completed; therefore an infinite sink is assumed to exist. No defects in work are allowed in this system; thus once a plate enters the system, it must pass through all of its elements, there being no provision for recycling or premature withdrawal. Additionally, no position swapping or priorities are allowed, as the queue discipline of first-come, first-served is maintained throughout. No machine breakdowns are recognized and personnel are assumed to work at a rate to insure applicability of the stated service time distributions. Finally, the amount of

time required to transfer a unit of throughput from the buffer storage to the process is included in the service time of the particular machine.

Now it is necessary to define the input and output variables, keeping in mind that a tradeoff is involved here; namely, if one chooses a great number of variables in order to describe in a very detailed way the system under consideration, the computational time will be longer and the computer simulation may even be rendered impossible because of insufficient computer memory capacity. Conversely, too few variables could result in an invalid model. A list of variables and their definitions is available in Appendix A.

The mathematical development of the expressions required to describe the probability mass and density functions and a random number generator are found in Appendix B. Included there are the appropriate computer flow graphs and FORTRAN programs that are used for this simulation.

Since the fixed increment time method is being employed, the mathematical expressions required to describe the flow of throughput in the system are not at all complex, and at most involve simple bookkeeping methods. Prior to referring to Appendix C, wherein are located the development of these expressions and the flow graphs for 3 basic systems, it is recommended that the following paragraphs regarding the advantages of a modular approach to computer programming formulation be read.

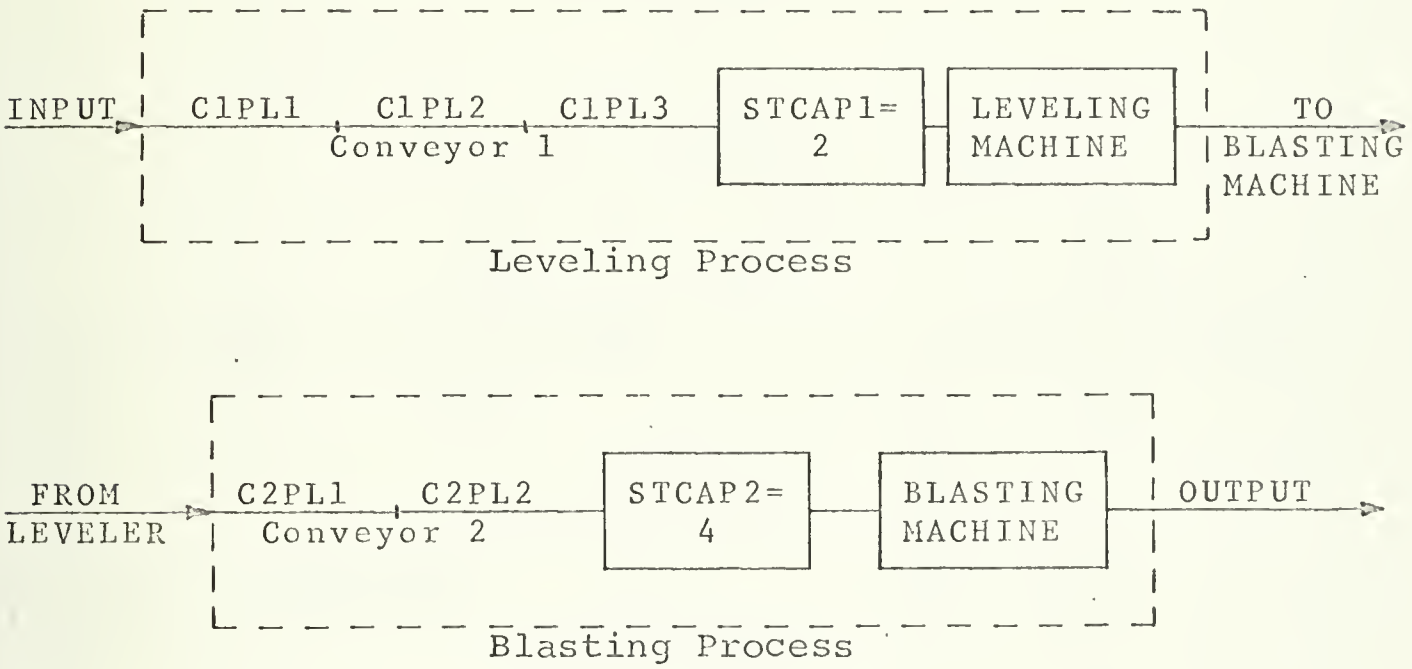
It is possible to consider the formulation of the computer program for the simulation of a production system in 2 ways, 1) by attenpting to write a program that describes the functioning of the entire system, or 2) by adopting a modular approach.

The modular approach is selected primarily because of 3 advantages that result. First, from a pedagogical standpoint the method is quite appealing. Second, in engineering systems models have been built up from available knowledge about the separate components. Designing a system model upward from identifiable and observable pieces is a sound procedure with a history of success [35]. Third, after constructing computer flow charts and programs for the modules, it is possible to generalize them into building block subprograms that can be used in the simulation of larger and more complex problems. Thus the modular approach offers the model builder the possibility of considerable flexibility.

Having developed a set of basic building block modules (see Appendix C) which make possible the rather realistic modeling of single throughput production processes, it is now necessary to consider the modifications that must be made to model the steel processing facility under consideration. This will be undertaken in 2 steps by 1) the interfacing of the single processes to form a multistaged production process, and 2) the incorporation of a variable throughput.

Consider now the manner in which the leveling process

and the blasting process can be combined to form a simple multistaged system. Figure 13 illustrates the subdivision of the system into the modules which represent each process, in addition to demonstrating some of the program variables. Employing the building block modules of type two from Appendix C, the flow graph for these processes is readily

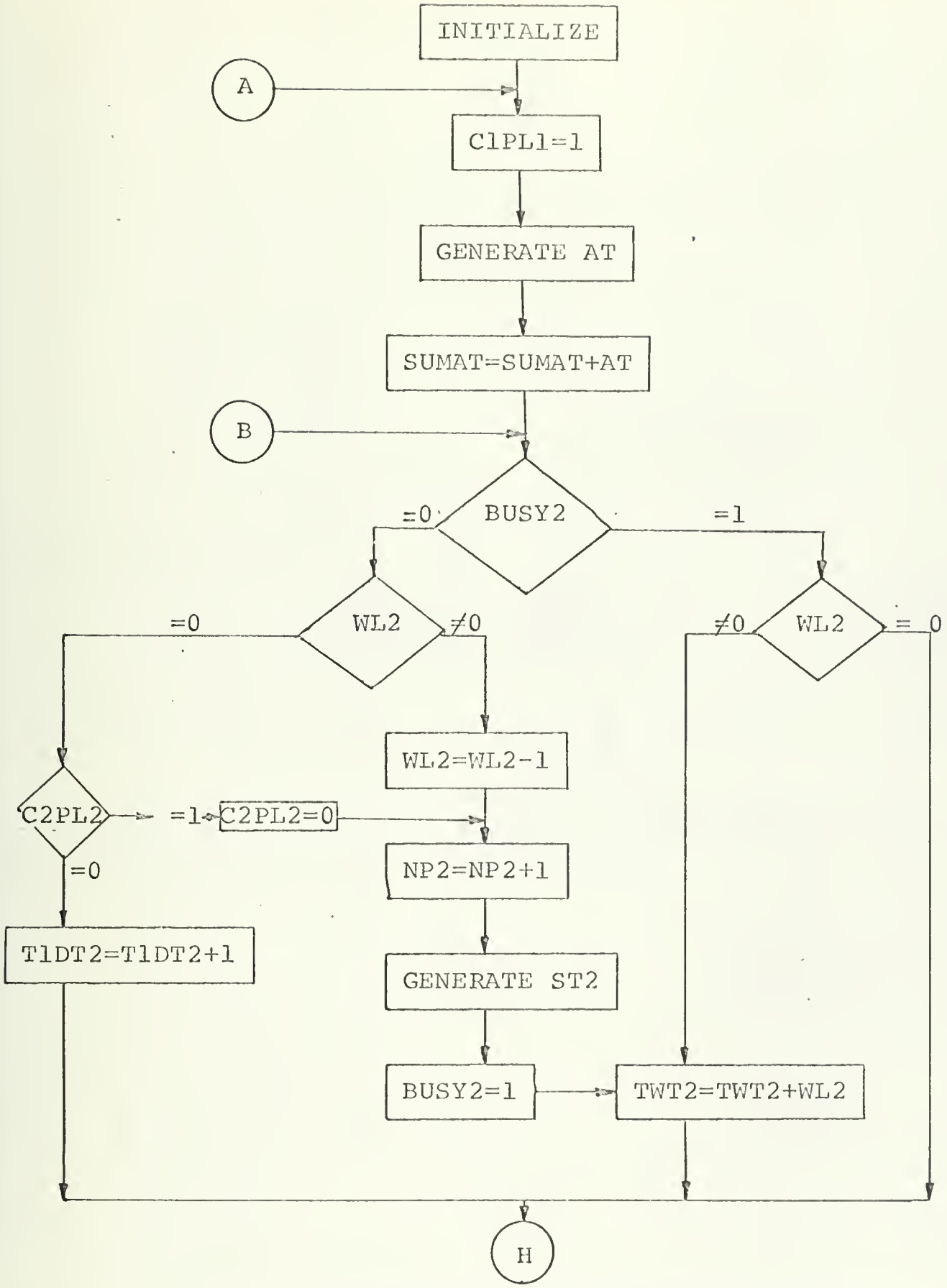


Subdivision of a Multistaged System into Individual Processes

Figure 13

constructed and is shown in Figure 14. From this flow graph the following general features of the building blocks are apparent.

The programs begin with a series of blocks that are common to all systems, beginning with the initializing statement



The Interfacing of Building Block Modules

Figure 14

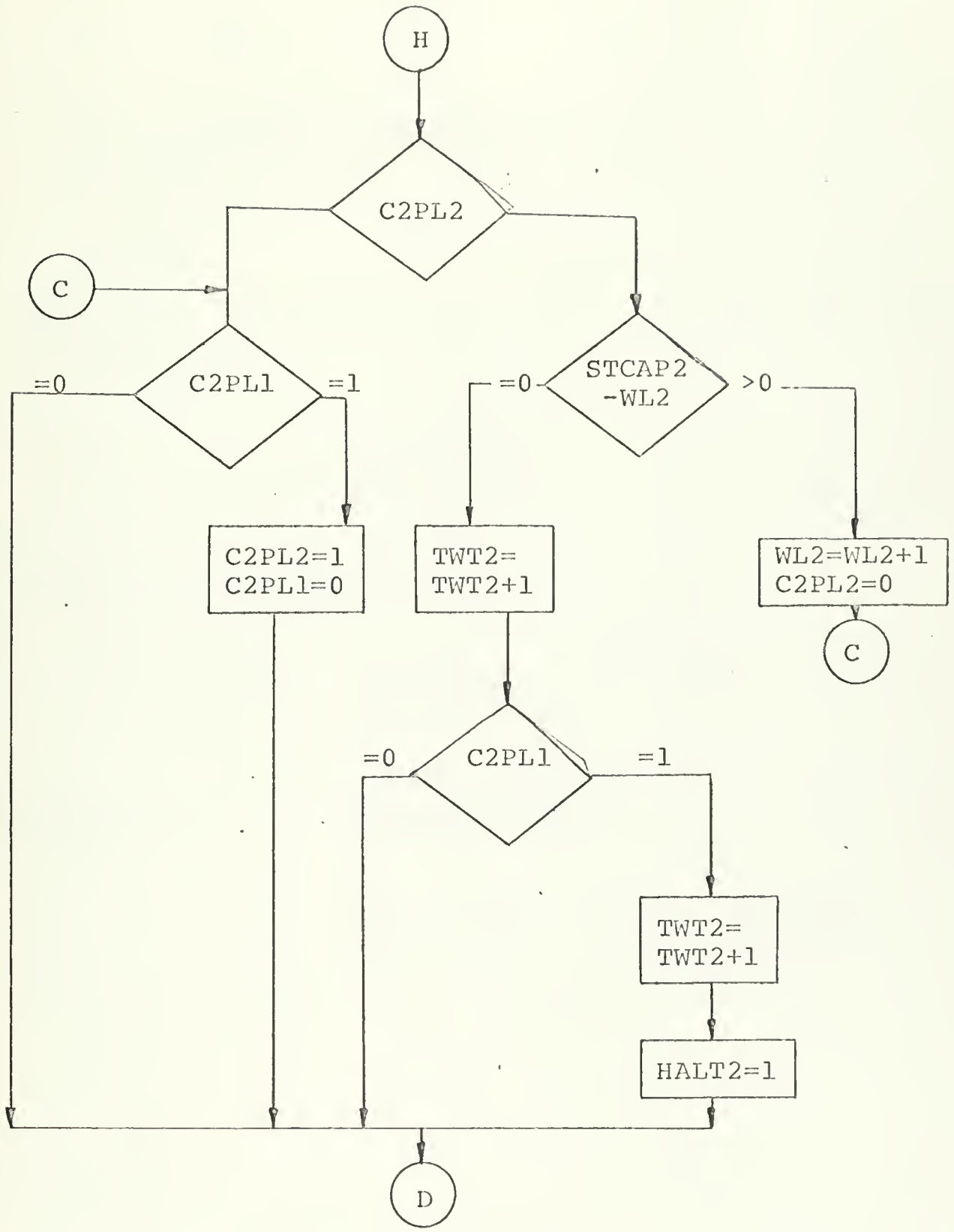


Figure 14 (continued)

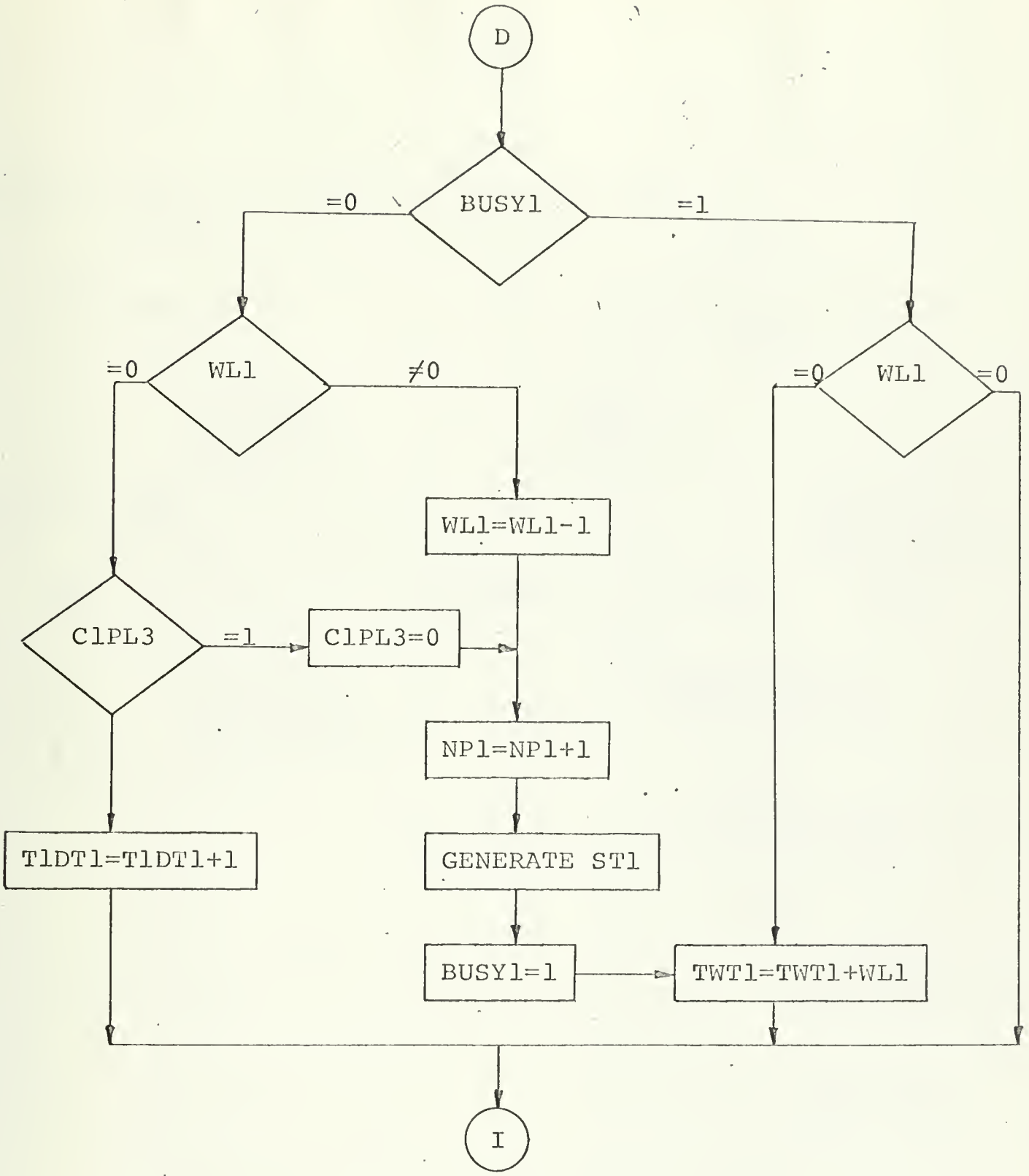


Figure 14(continued)

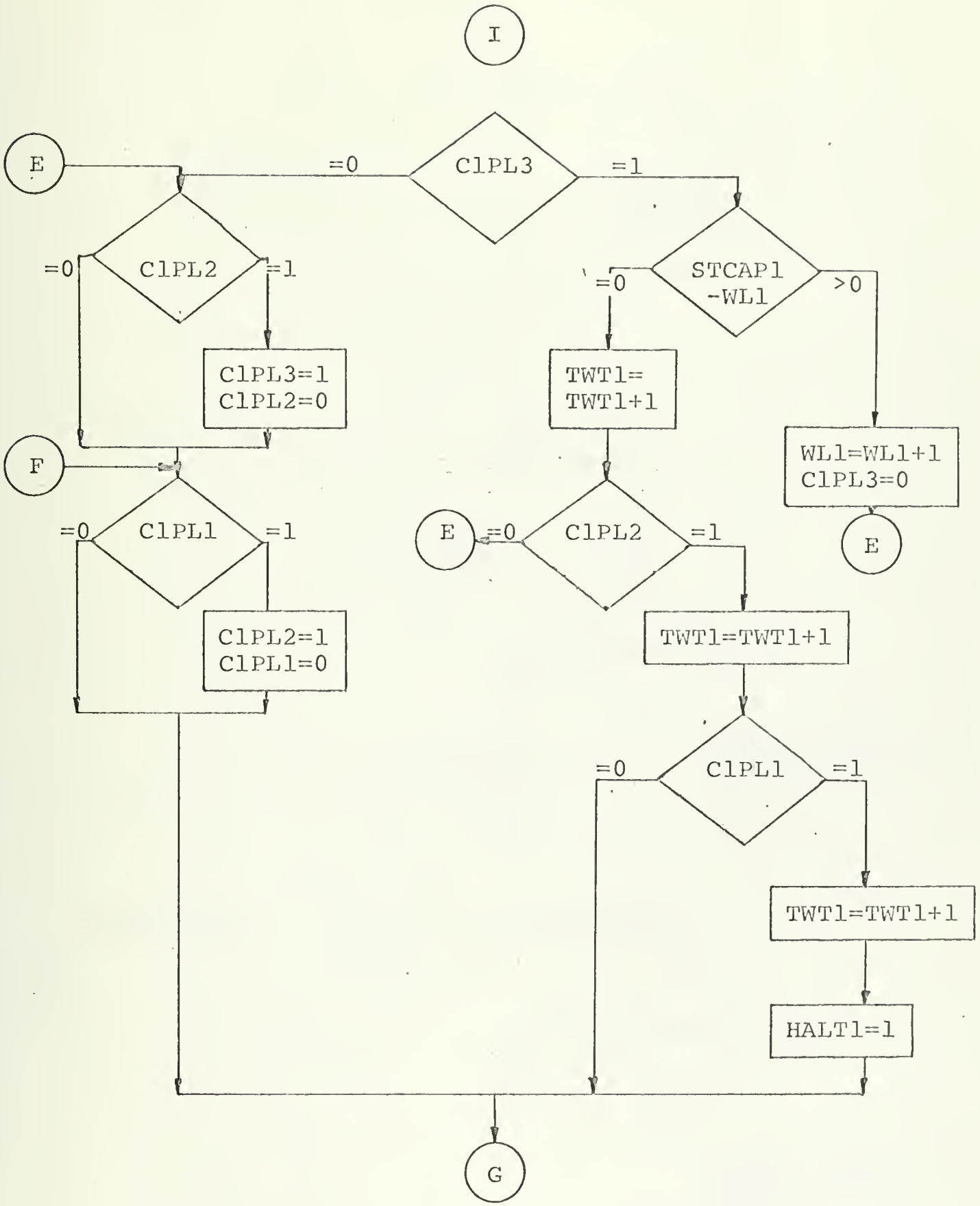


Figure 14 (continued)

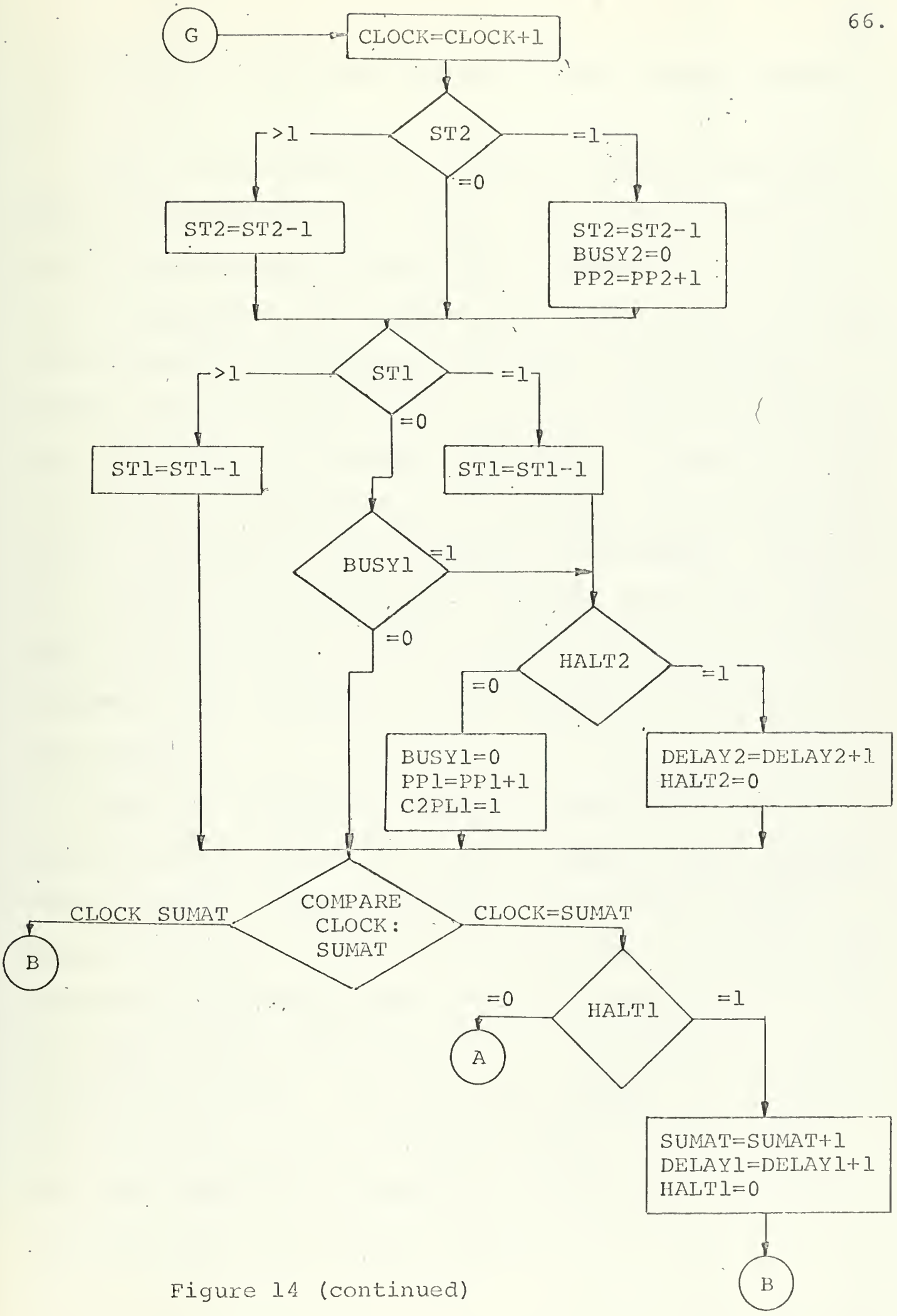


Figure 14 (continued)

and terminating with calculation of SUMAT, blocks 1 through 4.

Next, one notices that in order to describe the movement of throughput in the system, each process and each location within each process must be examined in reverse order of sequence. For example, the blasting process is the first process to be examined and within it the blasting machine, then the buffer storage area, and finally the conveyor positions are checked. Statements B through D of the blasting process and statements D through G of the leveling process correspond to this movement of throughput. Further, it is readily apparent from this portion of the flow graph that it is necessary to connect the building blocks at the statements that correspond to the physical interfacing of each process.

After all throughput movement is accounted for in a particular time period, the clock is updated, and then each service station status is revised. It is seen that the processes do not necessarily discharge their throughput upon completion of processing for there is a dependence on available space in the next position, as indicated by the appropriate HALT variable. In this case, though, it is assumed that the final process interfaces with an infinite sink, thus upon completion of servicing the plate is immediately discharged from the blasting machine.

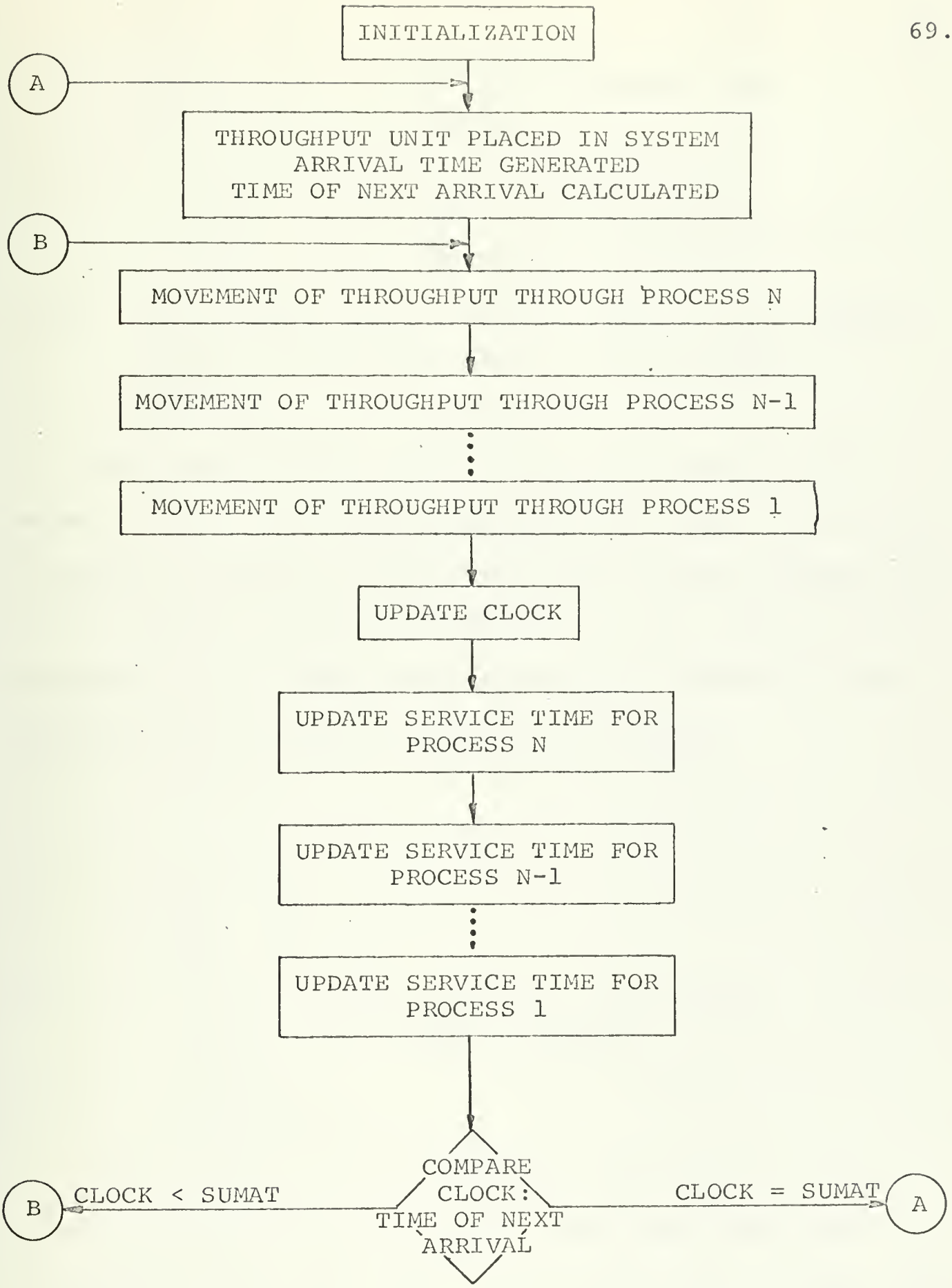
The final portion of the flow graph, beginning with the comparison of SUMAT and CLOCK is common to all models in which the possibility of system saturation exists. This group of blocks, in addition to those mentioned above, are illustrated in a general way in Figure 15.

Attention will now be focused on systems in which a variable throughput exists and the modifications that must be made to the building blocks of Appendix C in order to describe such systems.

It is obvious that some provision must be made for the identification of the throughput upon its arrival at the system and upon its arrival at the various service stations. The most convenient method in which this can be accomplished, without resorting to the identification of the throughput at every location in the system, is by the use of status variables that 1) identify the plate type by means of an integer variable and 2) identify, at each service station, the sequence number of the plate about to be processed.

In modifying the building blocks to accommodate the variable throughput for this particular problem, the following status variables are incorporated into the model:

I An integer variable used to identify the number in the sequence of plate arrivals. I = 1, 2, ..., m where m is the last plate to arrive in the system.



GENERALIZED COMPUTER FLOW CHART
FOR A SYSTEM COMPRISED OF N PROCESSES

Figure 15

PL(I) An integer variable that identifies the thickness of plate I by:

PL(I) = 1 if plate I is 1/2" thick
PL(I) = 2 if plate I is 3/4" thick
PL(I) = 3 if plate I is 1" thick

NPi An integer variable used to identify the sequence number of the plate about to be serviced at process i

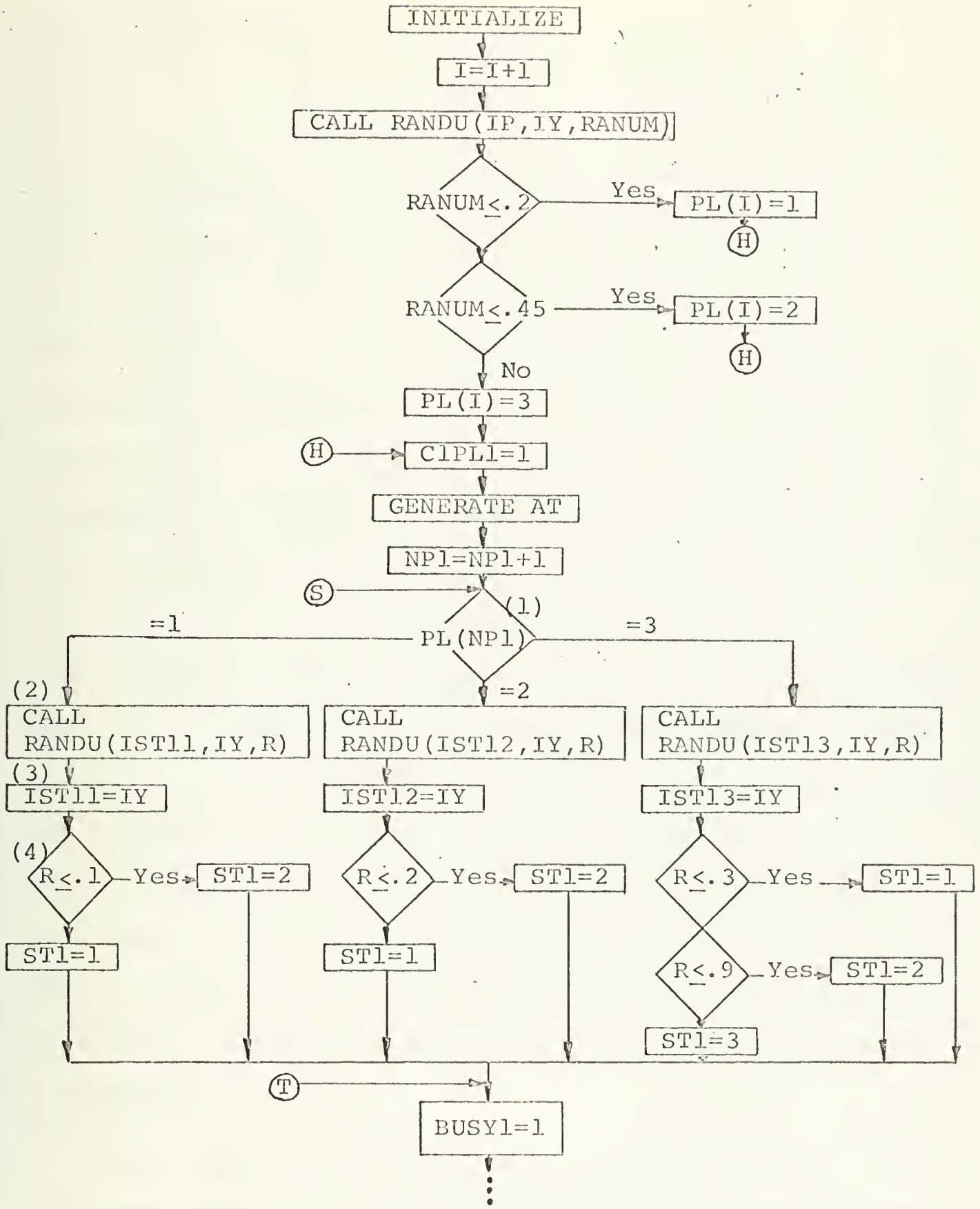
The computer flow graph in Figure 16 indicates the manner in which the leveling process flow graph was modified in order to conform to three plate types under consideration in the present analysis. In the first block, the new status variables and the first input argument for RANDU for each stochastic process are initialized as follows:

I = 0

NP1=0

IP=
IAT=
IST11=
IST12=
IST13=
} each a different odd integer of 9 digits or less

After incrementing the sequence number by one, the random number generator is used to determine the plate thickness in accordance with the specified PMF, in this case indicated in Figure 11. The arrival of the first plate in the system is then registered by statement H. Theprogram then



Leveling Process Flow Chart Showing Modifications Necessary to Accommodate a Variable Throughput

Figure 16

generates the arrival time of the next input unit, and proceeds using logic identical to that of the single throughput models until what was formerly identified as the GENERATE ST block is reached. All blocks between points S and T now perform the function of generating the appropriate service times for the leveler. Using the sequence number NP1, the thickness of the plate is identified by determining the value of PL(NP1) in block (1). By transferring control to the proper branch of the flow graph, the actual process of generating service times begins. For example, if the value of PL(NP1) is one, the branch corresponding to the 1/2" thick plate is here beginning with block (2). The random number generator is called directly since the service times for the leveler are elementary PMFs. Block (3) sets the next input argument of the random generator equal to the last integer output of the random number generator as required in Appendix B. Block (4) is used to determine in what range the real random number is located. If R is less than or equal to .2, a service time of two is generated for the leveler. If R is greater than .2 a service time of one is generated.

After the appropriate service time is generated, the flow chart again duplicates the logic of the single throughput model by setting the status variable BUSY1=1, moves the plates through the process, etc.

Thus by modifying the building block modules of Appendix C it was possible to write a computer program for the simulation of the steel processing facility. The computer program, as well as a discussion of its operation and restrictions and a macro flow chart are included in Appendix D.

E. Validation of Computer Program

The problem of validating a computer simulation model is indeed a difficult one because it involves a multitude of practical, theoretical, statistical, and even philosophical complexities. In general, however, two tests seem appropriate for this purpose. First, how well do the simulated output variables compare with known historical data. Second, how accurate are the simulation model's predictions of the behavior of the real system in future time periods?

Since the historical data in this analysis have all been hypothesized, such a method of validation cannot be now accomplished. However, it is noted that using deterministic queueing models and manual calculations, the model yields results that correctly describe such systems.

F. Design of Simulation Experiments and Analysis of Simulated Data

These final very important steps in the procedure for the planning of computer simulation experiments have little

physical meaning at this stage of the discussion since a hypothetical system has been the system under consideration for analysis. Due consideration is given these topics, though, in the chapter of this thesis entitled "Recommendations".

V. CONCLUSIONS

By employing the basic building block modules developed in Appendix C, a method evolved for the analysis of a steel processing facility, a facility typical of variable throughput production processes.

A computer simulation of realistic queueing models resulted from a method of analysis that meets the decision-maker acceptance criteria set forth in this thesis, namely, 1) the methodology is applied in an orderly and logical manner, 2) it is capable of realistically describing real world production systems by taking into account throughput transport, buffer storage, stochastically described arrival and service times, and variable throughputs, and 3) the character of the building block models is such that one can expect that the benefits of employing this methodology will certainly justify the time and effort required in the analysis of complex production systems.

Due to lack of time it was not possible to incorporate mathematical optimization techniques into the methodology. It is the opinion of the author, though, that relatively little additional effort would be required to demonstrate that the use of mathematical optimization techniques, such as those available in the IBM Mathematical Programming System [31], can be effectively utilized.

The most significant limitation of the models used in this methodology is due to the assumption of first-come, first-served queueing discipline. Although many production processes are adequately described by waiting lines of this type, it would be desirable to incorporate the ability to model other queueing disciplines as well; especially when considering the analysis of systems in which recycling or priority orders exist. It should also be mentioned here that as a result of this limitation, a degree of care must be exercised when modeling a system in which is located a multichannel process. When such a process exists it is necessary to redefine and keep track of the sequence numbers of the throughput upon completion of service and release from the process.

RECOMMENDATIONS

When contemplating those items which would have been desirable to accomplish, given time, a significant list develops since it appears that one could go on investigating endlessly variable throughput production processes. Regarding specifically the methodology addressed in this thesis, however, it appears that any future research should be undertaken adhering to the following recommended list of priorities.

First and perhaps most important, the computer models must be validated using real world data as input. Only by verifying that the building block modules are in fact capable of simulating actual production systems, and that they are capable of predicting future performance of the system, will further research be warranted.

Upon completion of validation, other simulation experiments can be planned, which no doubt will indicate that the capabilities of the building block models should be expanded so that their potential for application to actual production processes will be greater. Among the changes recommended for the building block modules are the following:

1) A provision to allow the incorporation of mathematical optimization techniques such as linear and dynamic programming.

2) A provision to allow the modeling of queueing

disciplines in addition to the first-come, first-served case that is now possible.

3) Provide other means of controlling the number of iterations that the simulation performs. Now the program runs until the required number of plates have been processed, but in some cases it may be more desirable to run the model by simulating a fixed time span, such as a shift, a week, etc.

Finally, even though the models developed may be somewhat limited, the potential benefits that may accrue from them are great. For example, one might use a building block model of an existing production system, and then change or modify process components to reflect a contemplated purchase of equipment. Simulation of such a model could be expected to yield information that is important to the decision-maker.

Thus, the last recommendation is that experienced decision-makers who have available a sufficient amount of data implement these methods when considered appropriate.

APPENDIX A

VARIABLE DEFINITIONS

Main Program Variables

AT — The arrival time of a unit of throughput at the first position considered in the system.

BUSYi — Status variable:

BUSYi =
- 0 if machine i is not processing throughput
- 1 if machine i is processing throughput

CiPLj — Status variable:

CiPLj =
- 0 if plate location j on the conveyor of process i is empty
- 1 if plate location j on the conveyor of process i is occupied

CLOCK — Simulated time measured in unit increments.

DELAYi — The total amount of time during which the arrival of another unit of throughput into process i is prevented due to the backlogging of throughput in the i^{th} or succeeding processes.

HALTi — Status variable:

HALTi =
- 0 if the arrival of an additional unit of throughput is not prevented from entering process i
- 1 if the arrival of an additional unit of throughput is prevented from entering process i

I — Subscript for do loop.

IS — Subscript for do loop

ISTij The entering argument for the random number generator (RANDU) when generating a stochastic service time at process j for plate type i, where i is equal to the present value of PL(I)

NPi Status variable used to identify the unit of throughput about to be serviced at process i.

PPij Number of units of throughput of type j serviced by process i.

STCAPi Buffer storage capacity, in units of throughput, at process i.

STi Length of time required to service the unit of throughput in the machine of process i.

SUMAT Total arrival time given by the sum of all the arrival times of the throughput units.

TIDTi Total amount of time that the machine of process i remains idle.

TA The arrival time (a real variable) which is generated by a given subroutine.

TS The service time (a real variable) which is generated by a given subroutine.

TWTi The amount of time that throughput units spend in a queue awaiting service by the machine of process i.

WLi The number of throughput units that are in the buffer storage area of process i awaiting service.

Subroutine Variables

EX The expected value of the random variable x

EXPENT A subroutine for generating exponentially distributed random variables (see Appendix B).

IX The entering argument for RANDU.

IY An output argument for RANDU.

NORMAL A subroutine for generating random variables that are from a normal (Poisson) distribution (see Appendix B).

RANDU A subroutine for generating random numbers (see Appendix B).

RANUM,R The random number generated by RANDU.

STDX The standard deviation of random variable x.

UNIFRM The subroutine that generates uniformly distributed random variables (see Appendix B).

APPENDIX B

GENERATION OF STOCHASTIC VARIATES

The probability distributions about to be discussed are in no way meant to be a comprehensive listing of possible stochastic processes. The distributions are presented solely to indicate the manner in which each was determined, the associated flow graph, and the resulting subroutine. The distributions considered are those which were found useful during the course of investigation of the pursued methodology.

Random Number Generator

Essential to the study of stochastic processes by computer simulation is the requirement for some method of generating random numbers. While it is impossible to produce truly random behavior by means of a computer program, many programs have been developed to behave in a "pseudorandom" manner.

The random number generator about to be described produces a real uniformly distributed random variable whose value lies between zero and one. In terms of probability density functions, the random variable x is a member of the following PDF.

$$f(x) = 1 \qquad 0 \leq x \leq 1$$

$$f(x) = 0 \qquad \text{elsewhere} \tag{B1}$$

The random variable x is determined by the application of congruential methods, which is based on the following recursive formula from number theory:

$$x_{i+1} = (ax_i+c) \bmod m \qquad (B2)$$

where x_{i+1} and x_i are integer variables and a, c and m are non-negative integer constants.

By selecting C=0, the commonly used multiplicative congruential method results. Further, on binary digital computer a modulus $m=2^b$ is used, where b = the number of bits in a word. Employing the multiplicative congruential method, IBM [50] has developed for use on the IBM 360 series computer the following subroutine for the generation of random numbers that are uniformly distributed between the values of 0 and 1.

```
C       RANDOM NUMBER GENERATOR
        SUBROUTINE  RANDU(IX, IY, YFL)
        IY = IX*65539
        IF(IY)  5,6,6
5       IY = IY + 2147483647 + 1
6       YFL = IY
        YFL = YFL*.4656613E-9
        RETURN
        END
```

The above random number generator is employed when necessary. The input argument for RANDU is IX, which when

first selected must contain any odd integer of 9 digits or less. After the first entry, IX should be the previous value of IY computed by this subroutine.

Probability Distribution Functions (PDFs)

Prior to considering the particular PDFs, a degree of repetition can be avoided by noting once some of the frequently used definitions that will be of use. Using the notation of Drake [17]:

A probability density function for the random variable x, $f_x(x_o)$ is given by:

$$\text{Prob } (a < x \leq b) = \int_a^b f_x (x_o)\, dx_o \tag{B4}$$

The cumulative distribution function (CDF) for the random variable x, $P_x (\leq x_o)$ is defined as:

$$P_x(\leq x_o) = \text{Prob } (x \leq x_o) = \int_{-\infty}^{x_o} f(x_o)\, dx_o \tag{B5}$$

The following properties of the CDF should be noted:

$$P_x(\leq \infty) = 1 \qquad P_x(\leq -\infty) = 0 \tag{B6}$$

$$\text{Prob } (a < x \leq b) = P_x(\leq b) - P_x(\leq a) \tag{B7}$$

$$\frac{d}{dx_o} P_x (\leq x_o) = f_x (x_o) \tag{B8}$$

The last property is especially useful, and maybe described in words by stating that the first derivative of the

cumulative distribution function with respect to its random variable is equal to the probability density function of the same random variable. This result is used frequently in the inverse transformation method for determining PDFs for random variables.

The expected value, mean, or first moment, of the random variable x is defined as

$$E(x) = \int_{-\infty}^{\infty} x\, f(x)\, dx \qquad (B9)$$

The variance, or the second moment about the mean, of the random variable x is given by

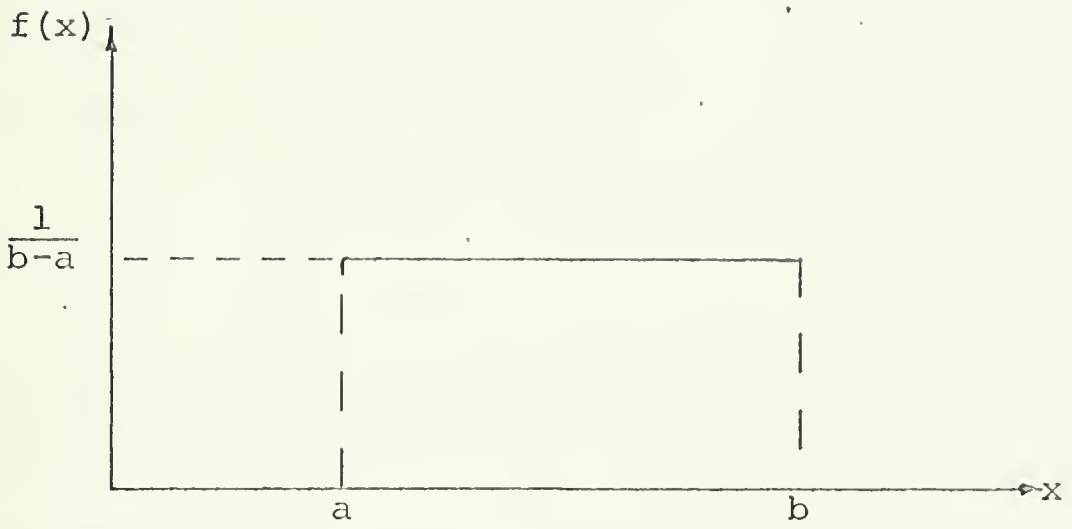
$$VAR(x) = \sigma^2 = E[(x-E(x))^2] = \int_{-\infty}^{\infty} [x-E(x)]^2 f(x) dx \qquad (B10)$$

Uniform Distribution

Perhaps the simplest continuous probability density function is the one that is constant over the interval (a,b) and is zero elsewhere. The principal value of the uniform distribution for simulation techniques lies in its simplicity and in the fact that it can be used to simulate random variables from almost any kind of probability distribution.

Mathematically it is defined as follows:

$$f(x) = \begin{cases} \frac{1}{b-a} & a < x < b \\ 0 & \text{elsewhere} \end{cases} \qquad \text{(B11)}$$



Uniform Distribution

Figure 1

The cumulative distribution, F(x) for the uniformly distributed random variable x is given by:

$$F(x \leq x_o) = \int_a^{x_o} \frac{1}{b-a}\, dx = \frac{x-a}{b-a} \qquad 0 \leq F(x) \leq 1 \qquad \text{(B12)}$$

The expected value and variance of the uniform distribution are given by:

$$E(x) = \int_a^b \frac{x}{b-a}\, dx = \frac{b+a}{2} \qquad \text{(B13)}$$

$$VAR(x) = \int_a^b \frac{[x-E(x)]^2}{b-a}\, dx = \frac{(b-a)^2}{12} \qquad \text{(B14)}$$

In actual practice the parameters of the uniform PDF, i.e., a and b, may not be directly known. Typically, though, one is able to obtain the expected value and the variance of the distribution. From these statistics one can readily derive the parameters of interest by simultaneously solving equations B13 and B14. The results are:

$$a = E(x) - \sqrt{3\ VAR\ (x)} \tag{B15}$$

$$b = 2E(x) - a \tag{B16}$$

To simulate the uniform distribution over some given domain (a,b) it is necessary to first obtain the inverse transformation of the CDF in terms of a new random variate as given by:

$$x_o = F^{-1}(r) \tag{B17}$$

Which can be summarized by saying that we allow the random variate r to be defined as:

$$r = F(x) = CDF(x) = \int_{-\infty}^{\infty} f(x)dx \tag{B18}$$

By solving equation B18 for x in terms of r, an explicit equation for x in terms of the random number r is obtained. Hence:

$$x = a + (b-a)r \qquad 0 \leq r \leq 1 \tag{B19}$$

The flow chart of this rather simple subprogram is given in the following figure.

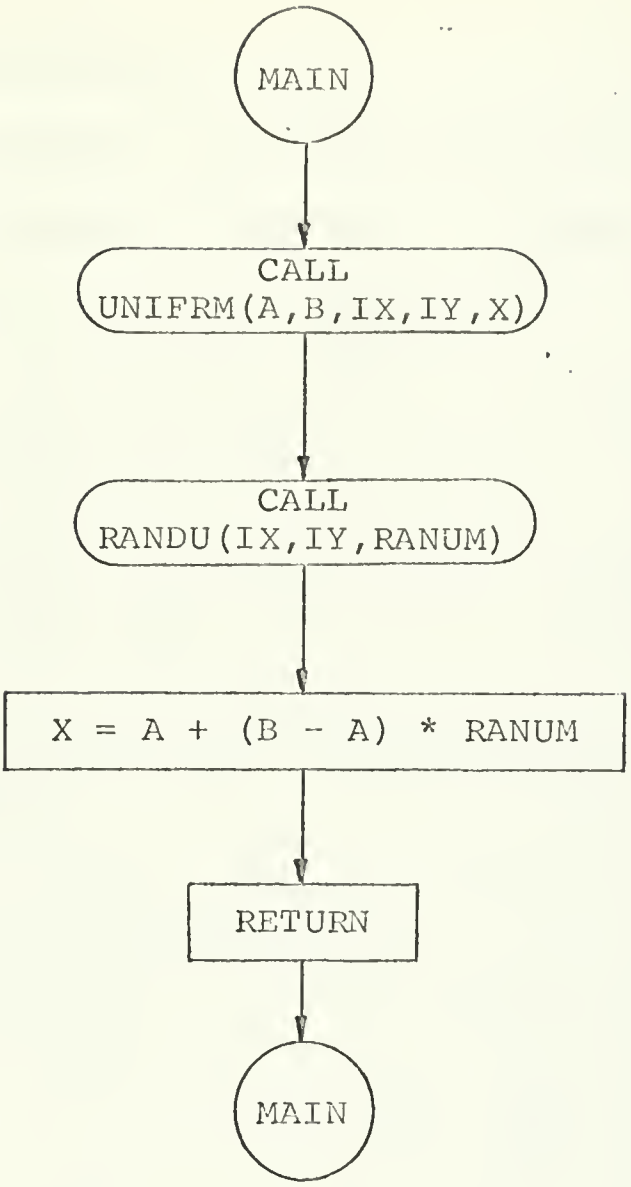


Figure 2

The uniform distribution subroutine is provided below:

```
C       UNIFORM DISTRIBUTION SUBROUTINE
        SUBROUTINE UNIFRM(A,B,IX,IY,X)
        CALL RANDU(IX,IY,RANUM)
        X = A + (B - A) * RANUM
        RETURN
        END
```

Exponential Distribution

A random variable x is said to have an exponential distribution if its density function is defined as:

$$f_x(x_o) = \alpha e^{-\alpha x_o} \qquad \text{(B20)}$$

The CDF of x is given by:

$$F(x<x_o) = \int_o^{x_o} \alpha e^{-\alpha x}\, dx = 1 - e^{-\alpha x_o} \qquad \text{(B21)}$$

and the expected value and variance of x are given by the following formulas:

$$E(x) = \int_o^{\infty} \alpha x e^{-\alpha x} dx = \frac{1}{\alpha} \qquad \text{(B22)}$$

$$VAR(x) = \int_o^{\infty} (x - \frac{1}{\alpha})^2 \alpha e^{-\alpha x} dx = \frac{1}{\alpha^2} = E^2(x) \qquad \text{(B23)}$$

The exponential PDF and CDF appear in Figures 3a and 3b respectively.

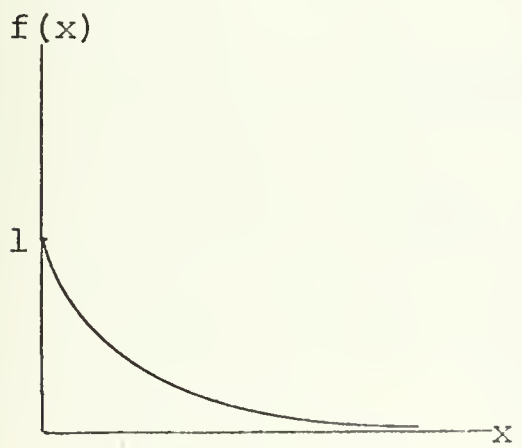


Figure 3a

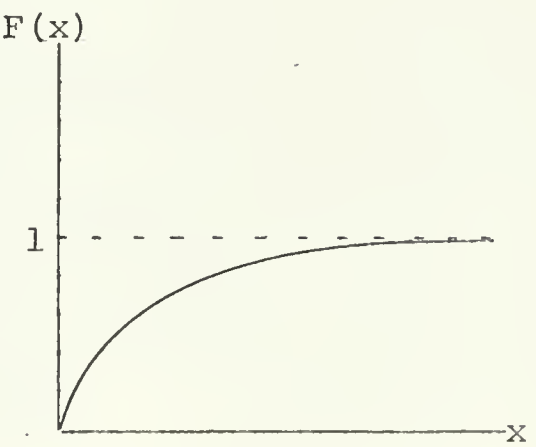


Figure 3b

The exponential is characterized by only one parameter, α, thus the distribution is readily obtained by noticing that:

$$\alpha = \frac{1}{E(x)} \tag{B24}$$

and then employing the inverse transformation technique as described in equations B17 and B18. Because of symmetry F(x) and 1 - F(x) are interchangeable. Therefore:

$$r = e^{-\alpha x} \tag{B25}$$

which then yields:

$$x = -\left(\frac{1}{\alpha}\right) \log_e r = -E(x) \log r \tag{B26}$$

Thus for each value of the random number r, a unique value of x is determined since r takes on only non-negative values ($0 \leq r \leq 1$). The random variable x will be exponentially distributed with an expected value of $1/\alpha$.

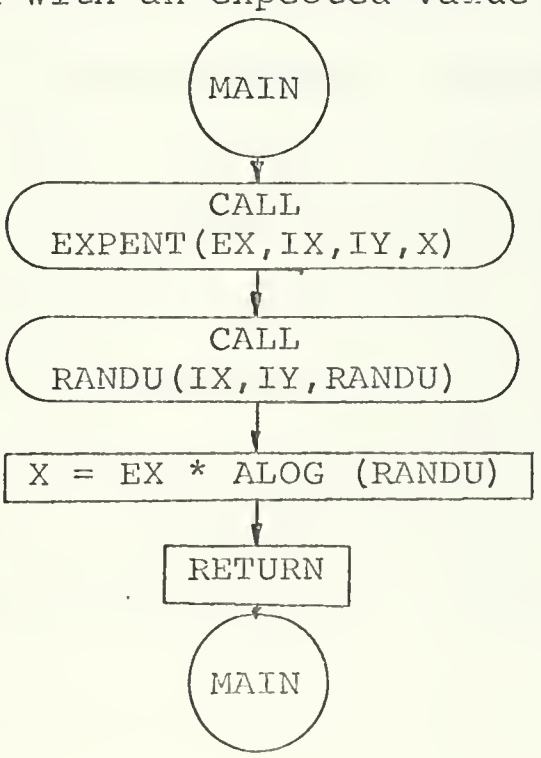


Figure 4

The exponential subroutine is given below.

```
C       EXPONENTIAL DISTRIBUTION SUBROUTINE
        SUBROUTINE EXPENT(EX,IX,IY,X)
        CALL RANDU(IX,IY,RANUM)
        X = EX * ALOG (RANUM)
        RETURN
        END
```

Normal Distribution (Gaussian)

The normal distribution is the best known and most frequently used probability distribution because of two reasons.

First, mathematical proof tells us that if a random variable can be defined to be the sum of n independent and identically distributed random variables, each of which has an expected value and a finite variance, in the limit as $n \to \infty$, the CDF of the new random variable will approach the CDF of the Gaussian random variable, regardless of the form of the individual PDFs in the sum. In addition, as long as a particular few of the number random variables do not dominate the sum, the identical distribution restriction can be relaxed so that for large numbers of random variables one is able to still use the Gaussian approximation expecting quite satisfactory results. Even the restriction on the independence of the random variables may be relaxed in certain cases.

Second, since many phenomena may be considered to be the result of a large number of factors, the central limit theorems, such as the example above, "are of great practical importance especially when the effects of these factors are either purely additive or purely multiplicative". [17]

The normal distribution is given by:

$$f(x) = \frac{1}{\sigma_x \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2} \qquad -\infty < x < \infty \qquad \text{(B27)}$$

where

$$\mu \text{ (the mean of the normal)} = \sum_{i=1}^{N} \mu_i \qquad \text{(B28)}$$

and

$$\sigma_x^2 \text{ (the variance of the normal)} = \sum_{i=1}^{N} \sigma_i^2 \qquad \text{(B29)}$$

The normal distribution is illustrated in Figure 5.

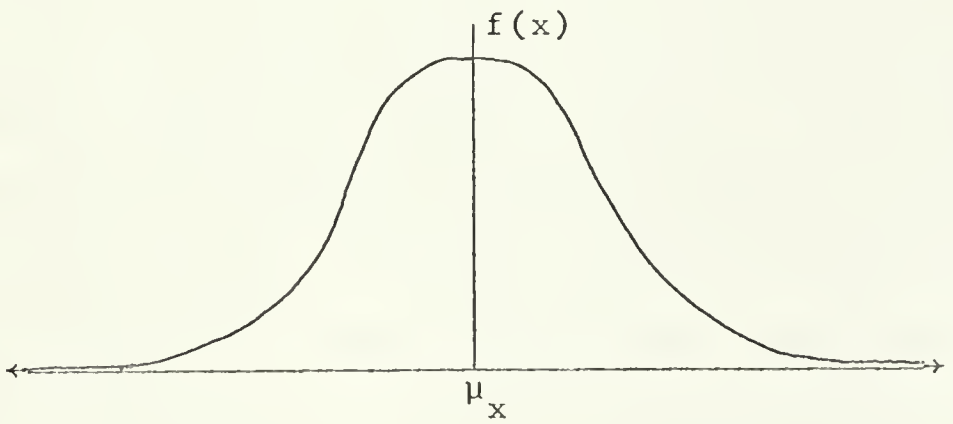


Figure 5

For the case where the distribution is centered about the origin and the variance is equal to one, the standard

normal results. Any normal distribution can be converted into the standard form by the substitution:

$$z = \frac{x - \mu_x}{\sigma_x}$$

Since the CDF for neither the normal nor the standard normal exists explicitly, one cannot use the inverse transformation technique to develop a normally distributed variate. By an interpretation of the Central Limit Theorem though, the following result is obtained [35]:

$$x = \sigma_x \left(\frac{12}{K}\right)^{\frac{1}{2}} \left(\sum_{i=1}^{K} r_i - \frac{K}{2} \right) + \mu_x \qquad \text{(B30)}$$

where K is defined as the number of independent uniformly distributed variables that are to be summed.

Equation B30 now provides a ready means by which a normally distributed variate may be generated on a computer. The value of K to be selected is determined by balancing computational efficiency against accuracy. The selection of K = 12 provides some degree of computational efficiency, while at the same time truncating the distribution at $\pm$ 6σ. The results have been found to be rather unreliable for values of x beyond three standard deviations, but a reasonably fast program results [34].

A flow chart for the normal distribution subroutine follows in Figure 6.

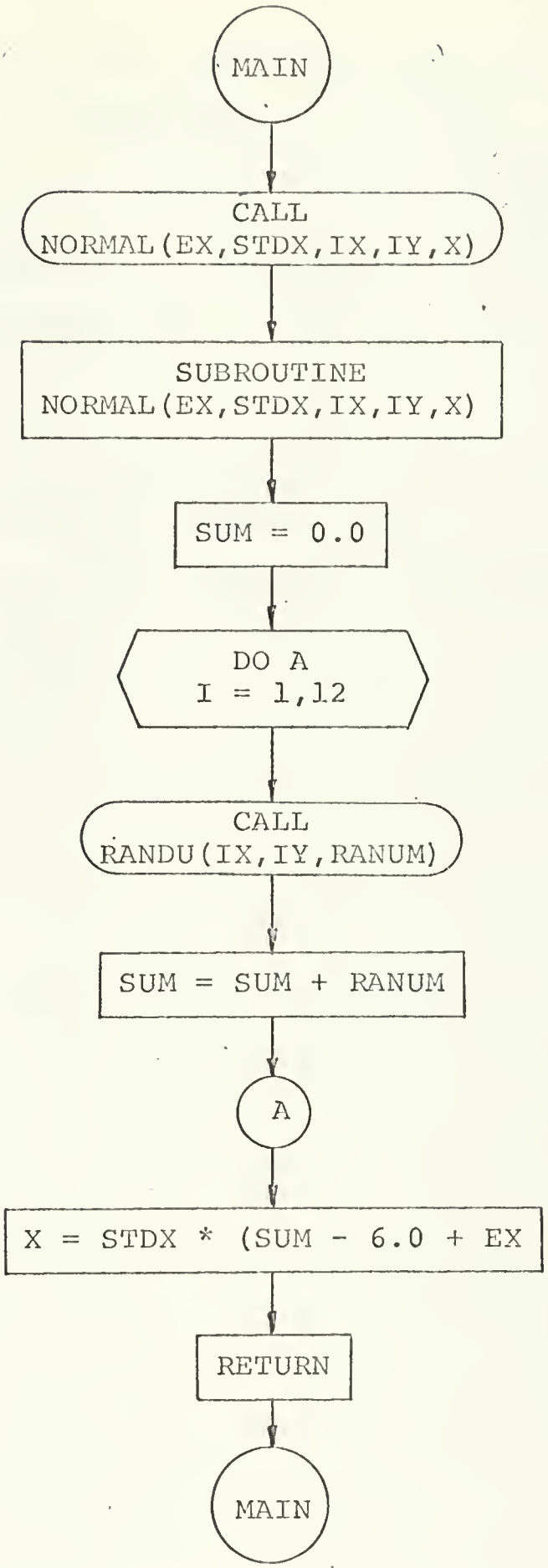

MAIN
CALL
NORMAL(EX,STDX,IX,IY,X)
SUBROUTINE
NORMAL(EX,STDX,IX,IY,X)
SUM = 0.0
DO A
I = 1,12
CALL
RANDU(IX,IY,RANUM)
SUM = SUM + RANUM
A
X = STDX * (SUM - 6.0 + EX
RETURN
MAIN

Figure 6

The subroutine for the generation of normally distributed variates is given below:

```
C       NORMAL DISTRIBUTION SUBROUTINE
        SUBROUTINE NORMAL(EX,STDK,IX,IY,X)
        SUMRAN = 0
        DO 1 I = 1, 12
        CALL RANDU(IX,IY,RANUM)
1       SUMRAN = SUMRAN + RANUM
        X = STDX * (SUMRAN - 6.0) + EX
        RETURN
        END
```

APPENDIX C

BASIC BUILDING BLOCKS FOR THE SIMULATION OF PRODUCTION PROCESSES

In formulating the following fixed time increment models of queueing systems, the procedures outlined in the flow chart of Figure 9 of the main text are used as guidance.

The variables (which, by the way, must all be integer for fixed time increment models) and the functional relationships of the system are defined in terms of a set of mathematical symbols. The resulting mathematical model is then converted into a computer flow chart, and ultimately into a computer program.

It is noted that the models neither adhere rigidly to the procedures of Figure 9, nor do they represent the most general case of the queueing phenomena being described. The reasons for this are:

1) To attempt to validate each basic building block model using real world data would be inefficient since it would be necessary to collect such data. Furthermore, the effort expended to collect such data would exceed that required for the solution of the models by manual calculation.

2) It is reasonable to expect that any systems under study would include a combination of queueing models.

Under such circumstances the data available would probably be associated with the whole system, and not necessarily with each particular component therein. Therefore the validation for the group of models would be based on the performance of the models interacting and simulating the system.

3) The models are not completely generalized because it is felt that to do so would cause some of the inherent characteristics of the building blocks to become lost or at least obscured.

4) The modification of the building block models to suit particular applications can be readily accomplished.

It is noted that a significant effort has been made to use terms and variable names that are logical, easily understood, and general in nature. In addition, as few restrictions as possible were imposed during the development of the building block models of production processes. When restrictions were necessary, such as when making assumptions regarding the queue discipline, attempts were made to adhere as closely as possible to realism. For example, with the steel processing shops in mind, a queue discipline of first-come, first-served is adopted, and it is assumed that the process times of the throughput include transit time from the buffer storage area to the service station.

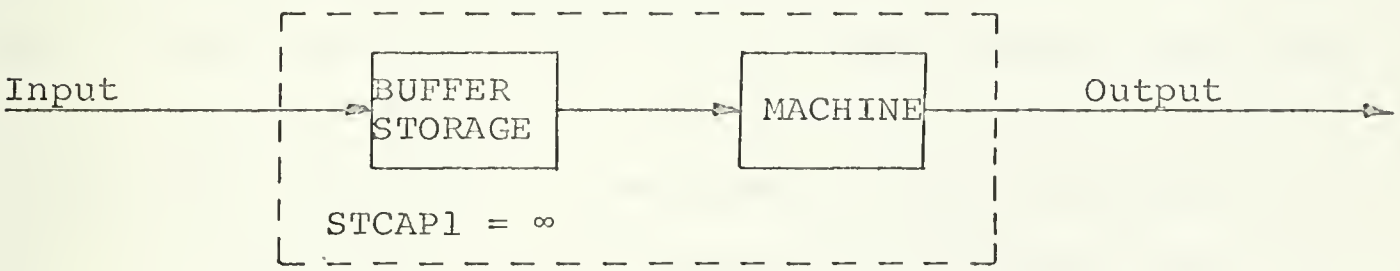
If the reader has not yet done so, it is recommended

that he consult Appendix A which contains a list of variable definitions that are used throughout the following pages. In addition, it is assumed that the reader has some knowledge of the terminology of queueing theory and is familiar with the form of FORTRAN notation.

Single Channel Queueing Models

MODULE 1

The most elementary building block that is of value in the simulation of production processes is the single channel queueing model identified as module 1 in Figure 1. (The buffer storage is indicated to insure consistency with the definition of waiting line of Appendix A.)



Module 1

Figure 1

This most basic model is characterized as having a single server (machine), a single throughput, an infinite storage capacity in the buffer storage area, and a first-come, first-served (FCFS) queueing discipline.

The variables for the mathematical model are defined as:

AT The interval of time between the i^{th} arrival and the $(i-1)^{st}$ arrival at module 1.

ST The service time of the i^{th} arrival in the machine of module 1.

SUMAT The total arrival time when the $(i+1)^{st}$ unit arrives at the service station.

WT The amount of time the i^{th} unit spends in the buffer storage area waiting to enter the service station (machine).

IDT The amount of time the service station remains idle waiting for the i^{th} unit to arrive at the machine.

TWT The total time spent waiting by all i units when the i^{th} unit enters the service station.

TIDT The total that the service station has been idle when the i^{th} unit arrives.

WL The number of units in the buffer storage area awaiting service.

CLOCK The simulated time measured in unit increments.

i A subscript used to identify a particular unit $i = 1, 2, \ldots, m$.

When the first input arrives at the service station, i.e., when i = 1, the following conditions are assumed to exist.

$$SUMAT = AT = 0$$

$$TIDT = 0$$

$$TWT = 0$$

$$WL = 0$$

$$CLOCK = 0$$

The computer flow chart for module 1 is shown in Figure 2. The first block indicates that the initial conditions have been set. In block 2 the length of the queue is increased by one unit indicating that the first unit has arrived. An arrival time is then generated by the appropriately called subroutine, thus the time interval until the arrival of the next unit is obtained. In block 5 the status variable associated with the service station, in this case BUSY1, is checked to determine if the service station is occupied or empty. If BUSY1 = 0 the service station is empty, so then the waiting line is checked to determine if there is a unit awaiting service. If the waiting line is not empty, i.e., WL > 0, the waiting line is decreased by one unit as indicated in block 9. The status variable for the service station must then be changed to reflect the movement of one unit from the waiting line to the service station, hence BUSY1 = 1. A service time now generated for the unit that has entered the service station results by calling the appropriate subroutine. One unit of waiting time is then added to the total waiting for each unit that is in the waiting line. If in block 6 it is found, however, that the waiting line is empty, a unit of idle time would be added to the total idle time for the system.

If the service station was occupied when block 5 was checked, the waiting line would then be checked to see if there are units awaiting service. If there are, one unit

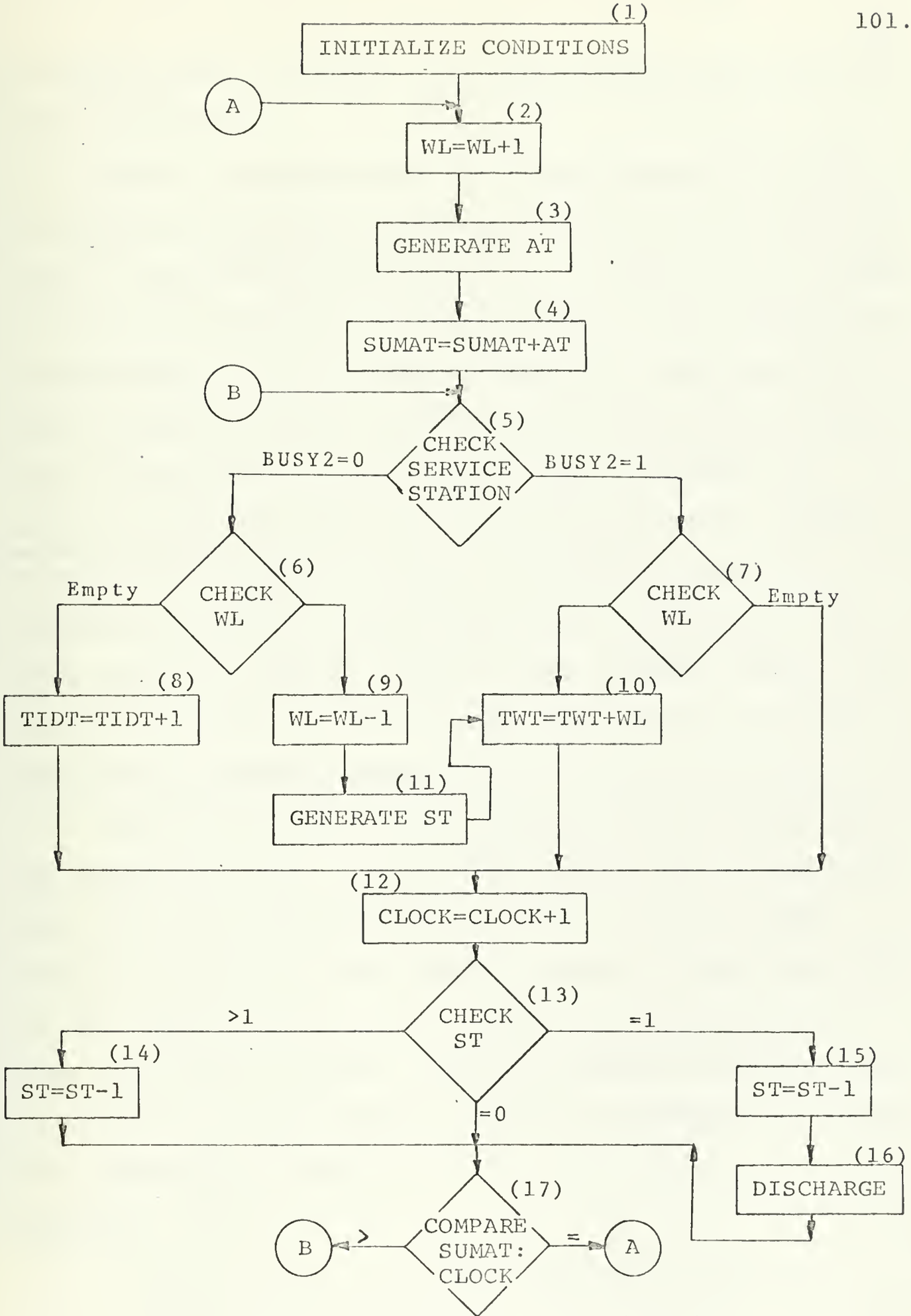


Fig. 2 Computer Flow Chart for Module 1

of waiting time is added to the total waiting time for each unit in the queue.

Having considered all the logical possibilities for the status of the service station and the waiting line, the clock is then advanced in block 12. In block 13 the status variable indicating the amount of service time remaining to be performed, ST in this case, is checked. (The service station check in block 5 depends entirely upon the service time in block 13.) If the service time is greater than unity, the service time is decreased by one unit. If the service time is equal to unity, the service time is decreased by one unit, and then the unit is discharged. At this point, the service station status variable, BUSY1, is again set to zero. If the service time is equal to zero, the service station is empty.

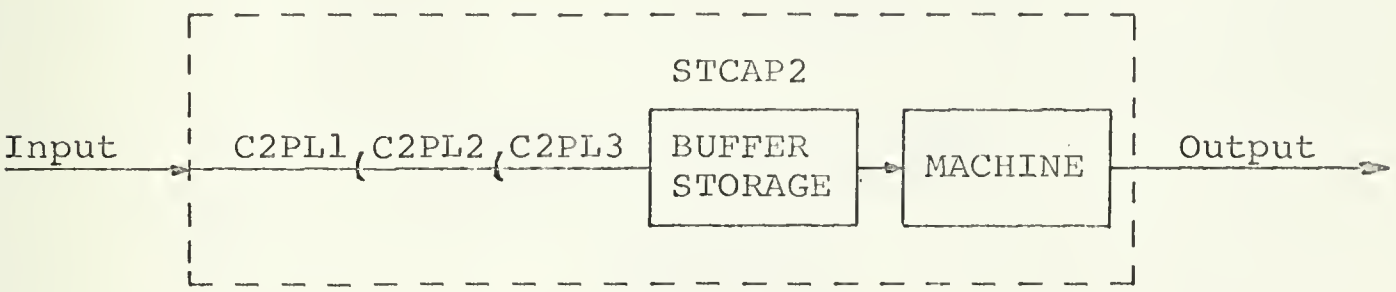
The clock time is then compared with the total arrival time in block 17. If the clock time equal to the total time, then control is shifted to block 2 where an additional unit arrival takes place. However, if the clock time is less than the total arrival time, the time has not yet come for another arrival, therefore control is shifted back to block 5 to begin another iteration of updating all events and advancing the clock.

MODULE 2

Although there is value in exercising the single

channel queueing model in module 1 since it affords one the opportunity to become familiar with the sequence control problem, it is rarely of any practical use.

For most production processes, the first single channel queueing model of any practical value is one which possesses character similar to that described in module 1, with two significant exceptions: 1) the realistic model provides for a conveyor, or other transporting device to handle the throughput between the interface of the input with the process and the buffer storage area; 2) the buffer storage area is of a finite capacity. Such a model, with a three plate length conveyor, is shown in Figure 2.



Module 2

Figure 3

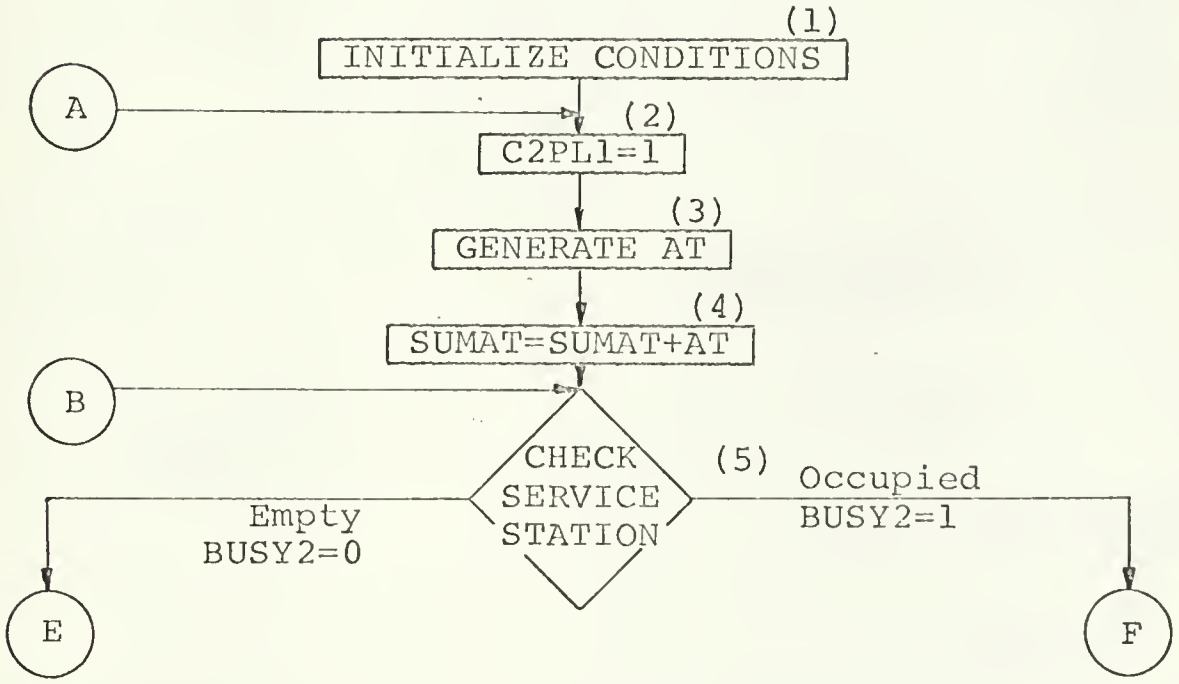
The variables for the mathematical model of module 2 are the same for module 1, with the following two exceptions:

TWT The total amount of time that throughput units spend in a queue waiting service. In the event that the buffer storage area is filled to capacity, TWT will include the amount of time that throughput units wait to enter the buffer storage area.

DELAY The total amount of time during which arrivals are prevented from entering the process due to all locations being occupied.

There are additional status variables that are required in the computer program. These are indicated in the flow chart of Figure 4 and are defined in Appendix A. The first block again sets initial conditions for the system. The system is assumed to be empty at the start of the simulation, therefore the initial conditions are:

SUMAT = 0	DELAY2 = 0
TIDT2 = 0	HALT = 0
TWT2 = 0	BUSY2 = 0
WL2 = 0	C2PL1 = 0
CLOCK = 0	C2PL2 = 0
	C2PL3 = 0



Computer Flow Chart of Module 2

(continued)

Figure 4

Fig. 4 continued

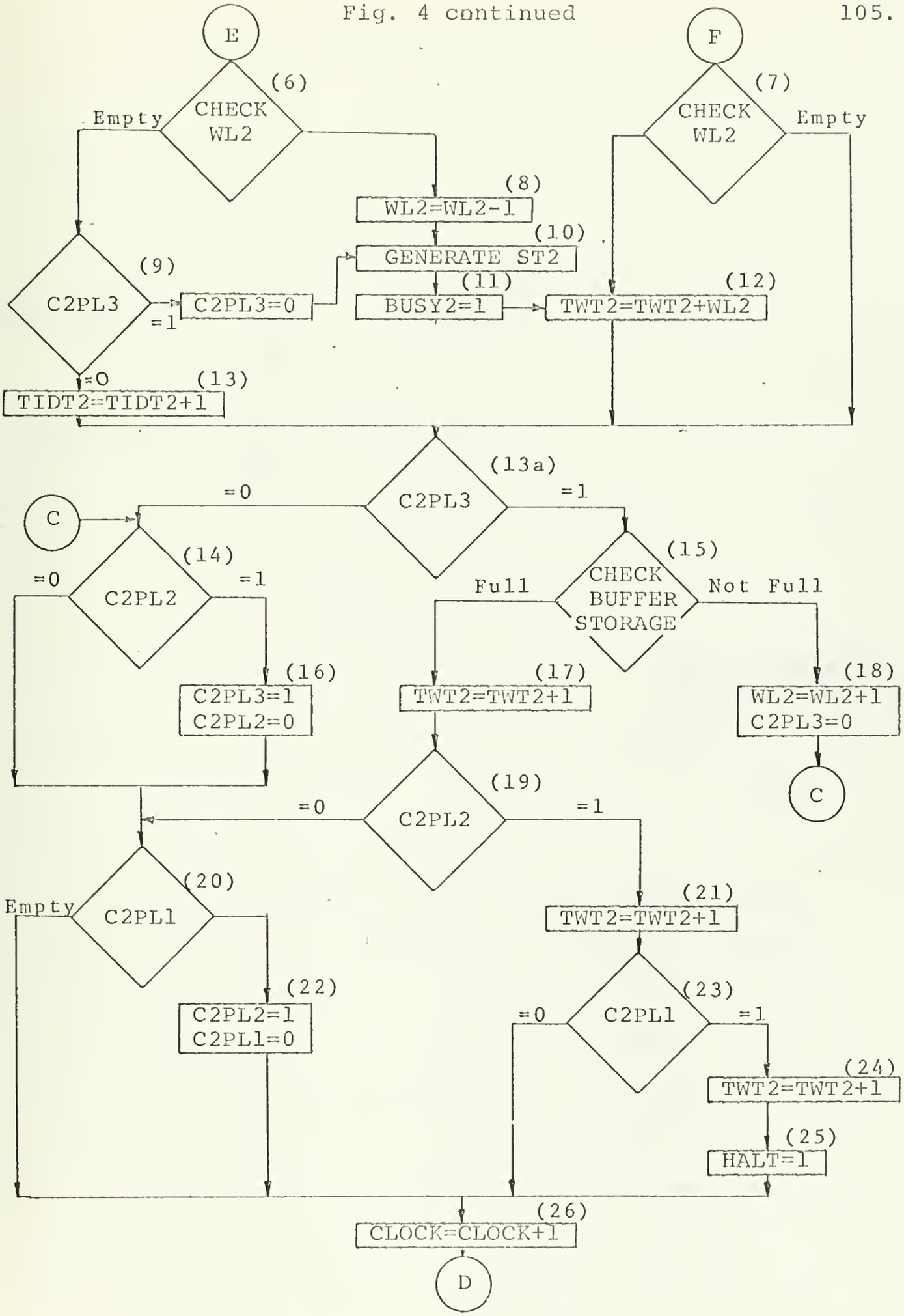
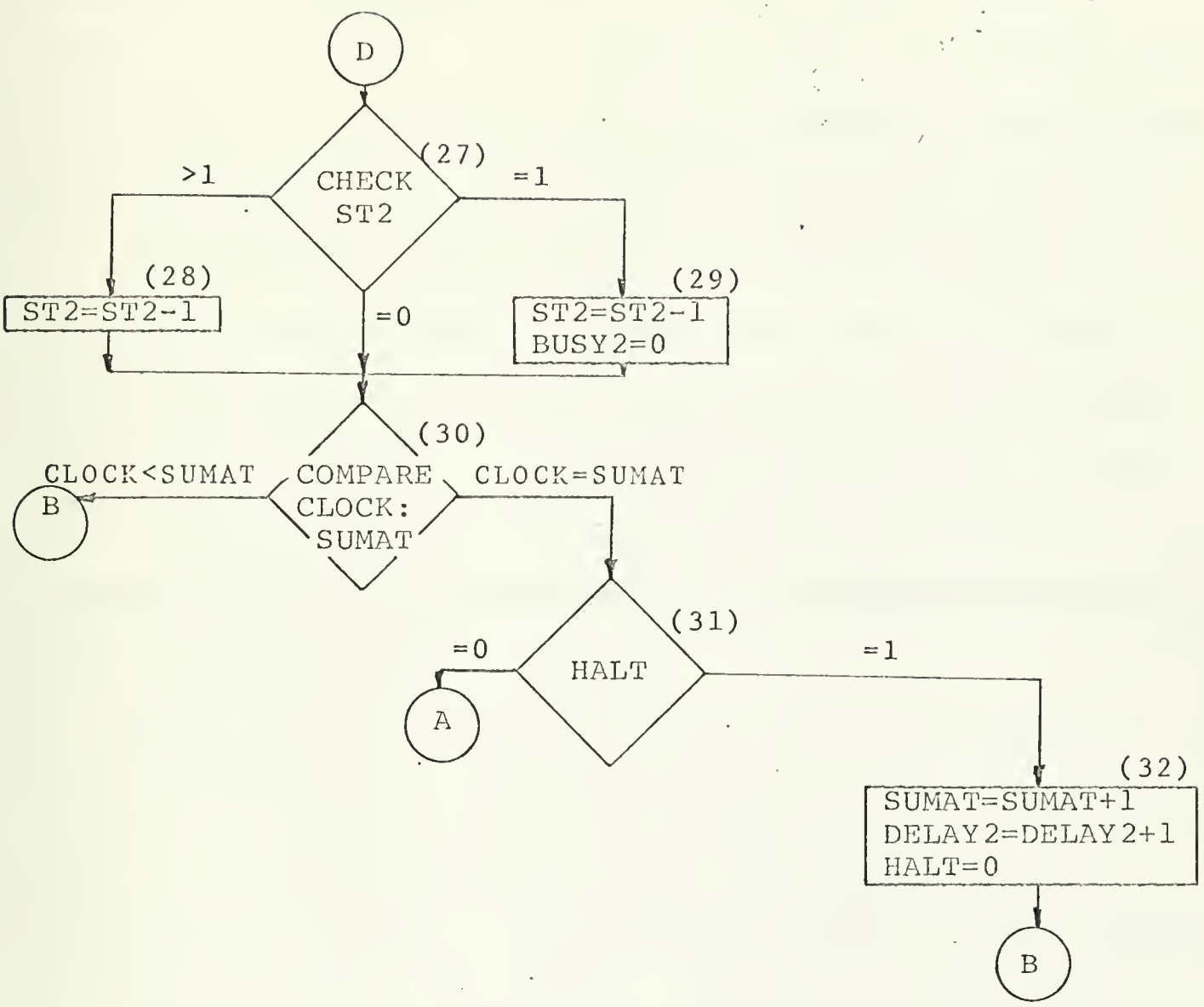


Fig. 4 continued



On the other hand, if the conveyor position immediately preceding the storage area is occupied, block 13, and the buffer storage area is filled to capacity, a series of checks are made to determine if the system is saturated and if any future unit arrival will be delayed. The path through blocks 13, 15, 17, 19, 21, 23, 24 and 25 is completed during system saturation. The total waiting time is first increased by one unit in block 17 if the unit in C2PL3 must wait to enter the storage area. If C2PL2 is also

occupied, an additional unit of waiting time is added in block 21. The final unit is added to the total waiting time when block 24 is reached. The control variable HALT is then set equal to one to indicate that the system is saturated.

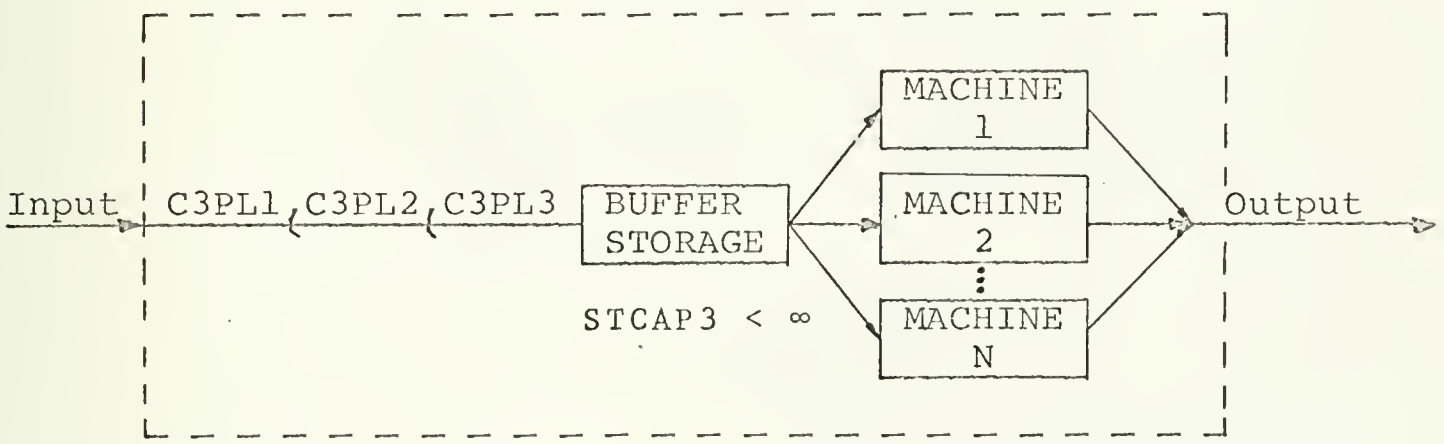
Of course, if after finding the buffer storage area filled to capacity, having already determined that C2PL3 is occupied, block 19 can result in finding C2PL2 empty, such that system saturation is avoided and the transfer of control to block 20 is effected. Even if C2PL2 were also occupied, it is still possible to avoid system saturation if C2PL1 is found empty in block 23.

Having considered the logic of a "typical" conveyor, the clock and service time calculations are made as for module 1. Upon comparing the clock time with the total arrival time in block 30, and determining that CLOCK = SUMAT, the last deviation from module 1 logic takes place. The status variable HALT is checked in block 31. If the value of this variable indicates that the system is saturated, the arrival time of the next arrival is increased by one unit, the total delay of the system is increased by one unit, and HALT is set back to zero as shown in block 32. Transfer then takes place to block 5 where another iteration of updating takes place. If in block 31 HALT indicates that the system is not saturated, control is transferred to block 2 where another unit is allowed to enter the system.

Multichannel Queueing Model

Module 3

The model to be considered represents a system composed of a conveyor, buffer storage area of limited capacity, and n service stations operating in parallel (Figure 5). Input units arrive at the system, transit the conveyor and buffer storage area, and then are accepted for service according to a first-come, first-served queueing discipline. The service time for each of the n service stations is a stochastic variate, with each station having its own probability distribution for service time.

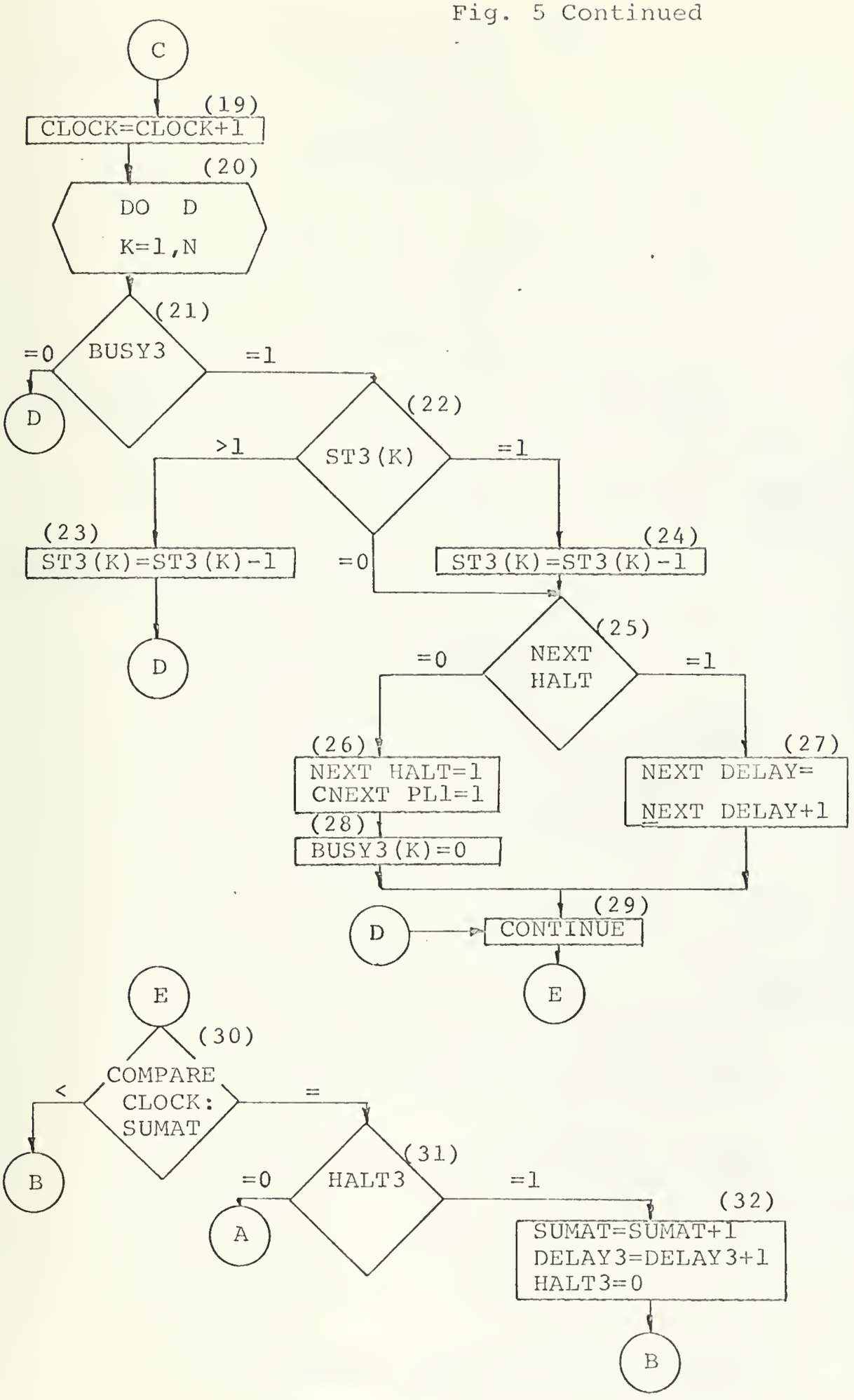


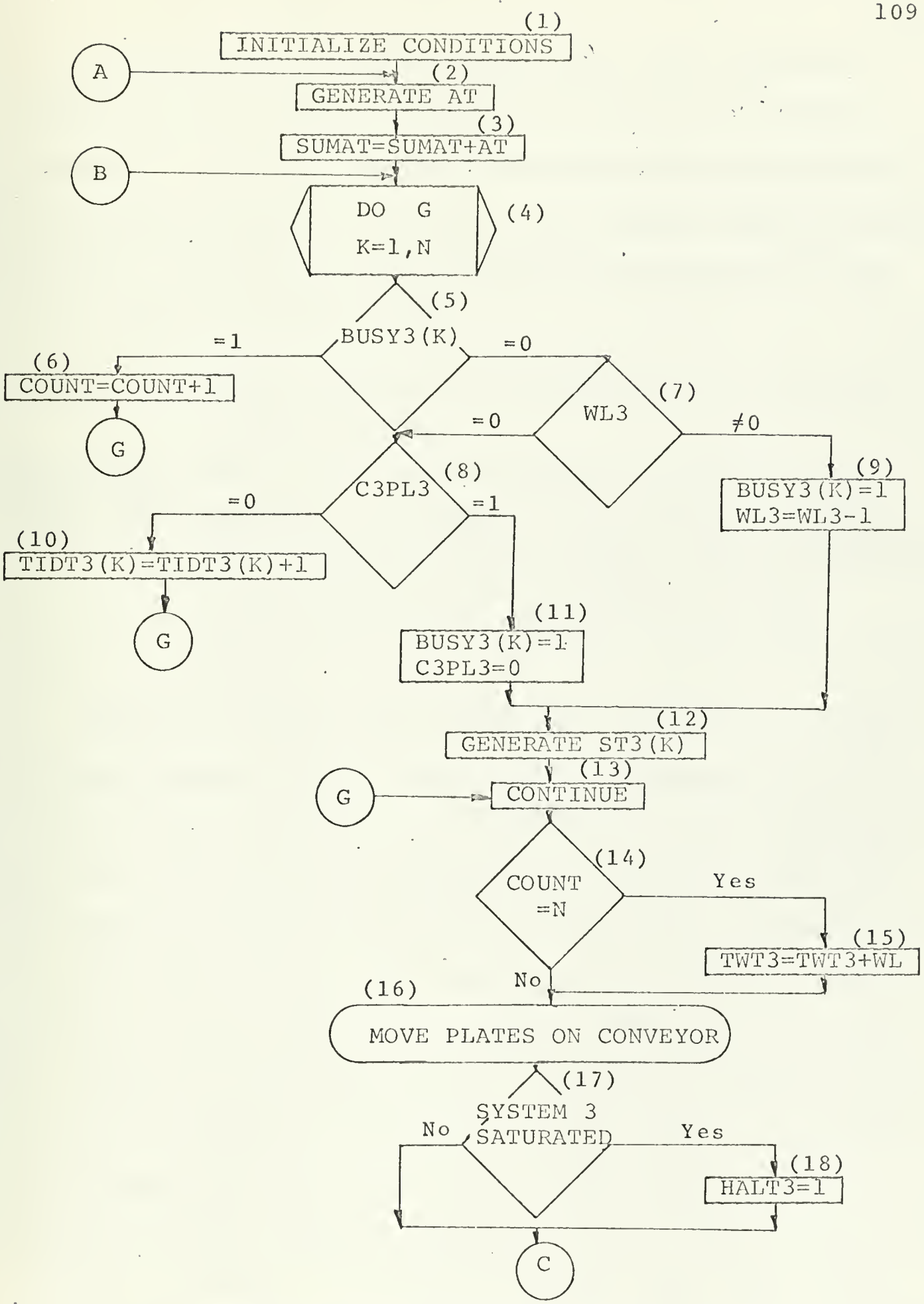
Module 3

Figure 5

Upon reaching the buffer storage area, the n service stations are checked to determine whether any are vacant. If all stations are occupied, waiting time occurs until one station becomes vacant. When there are no units available to enter a service station, idle time for that station occurs.

Fig. 5 Continued





Computer Flow Chart of Module 3

Figure 5

The variables required to describe this system are similar to those used in the preceding modules, with the exception that the subscripted variables are written in the same form as FORTRAN variables. For example, $BUSY3_i$ becomes BUSY3(I). The definitions of variables mentioned for the first time are listed below:

COUNT — Status variable for determining the existence of waiting time.

NEXT HALT, NEXT DELAY, CNEXT PL1 — NEXT is used as an adjective to describe the appropriate variable in the following process, if a following process exists. Hence CNEXT PL1 is the variable that corresponds to plate location one on the conveyor of the next process.

Blocks 1 through 3 are identical in function to those of the preceding models. For reasons of programming efficiency, block 4 indicates the beginning of a do loop which performs the function of checking each service station to determine if another unit can be accepted for processing, or if idle time occurs. Upon finding service station K empty in block 5, the buffer storage area is checked to determine the length of the waiting line. If there is a waiting line, a unit is accepted for servicing and the queue is decreased accordingly in block 9. If the waiting line is empty, though, the last conveyor position prior to entering the storage is checked. If it too is empty, idle time occurs and is recorded in block 10. If the last conveyor position

however, the input unit is accepted for service. The appropriate service time is then generated in block 12. This procedure then continues until all the machines of the process have been checked. If it should happen that all machines are busy, as determined in block 14, the total waiting time is appropriately adjusted.

Block 16 indicates the movement of throughput units on the conveyor and is accomplished in a manner identical to that described in the single channel queueing model, module 2. By using this procedure, a determination of system saturation is obtained in block 17.

After advancing the clock, it is necessary to adjust the service times of all the machines in the system that are presently servicing throughput. This is most easily accomplished by use of another do loop beginning in block 20. If a particular machine has completed servicing the throughput unit, the availability of a location in which to discharge the unit is determined in block 25. If the next process is saturated, i.e., NEXT HALT = 1, delay time is charged to the next process in block 27. If there is room for a unit of throughput in block 25, the status of the first plate location on the following process conveyor is then revised, as is the NEXT HALT variable. In block 28 the discharge of the unit from the service station is recorded.

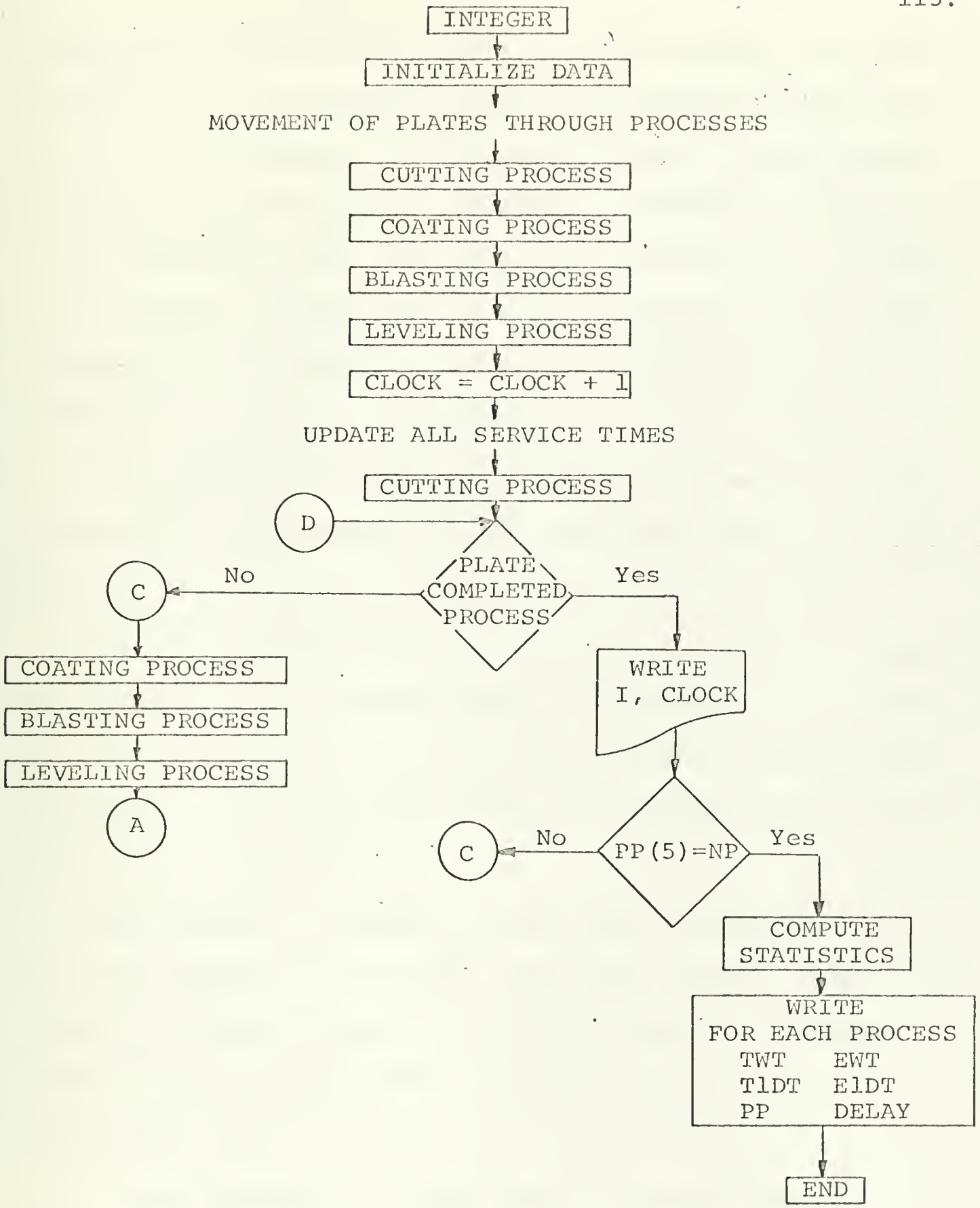
The procedures involved in comparing clock time with total arrival time and the resulting transfer of control to the next iteration, blocks 30 through 32, are identical to those of module 2.

APPENDIX D

SIMULATION OF A VARIABLE THROUGHPUT PRODUCTION SYSTEM STEEL PROCESSING FACILITY

This appendix provides information concerning the actual computer program utilized in the simulation of the steel processing facility described in the main text of this thesis.

The macro flow chart, Figure D-1, illustrates the general form of the computer program for the simulation of the steel processing facility. The first block indicates that the variables of the problem are declared to be integer, a requirement of the fixed time increment method employed. Next, the initial values of all data, i.e., counters, parameters, plates to be processed, etc., are set. Commencing with statement A the movement of the plates through the system is described (note that the processes are examined in the reverse order of their actual sequence). Upon completing the movement of the throughput in all processes, the clock is incremented, followed by the appropriate revisions of the service time of all units of throughput presently occupying a service station position. At statement D, if it is determined that a plate has completed servicing at the last process, the command is given to write the sequence number of the plate, I, and the time, CLOCK, at which plate I completed its processing in the system. A



MACRO FLOW CHART OF COMPUTER PROGRAM
FOR THE STEEL PROCESSING FACILITY

Figure D-1

comparison is then made between the total number of plates that have been processed by the system, PP(5), and the number of plates required to be processed, NP. If all plates requiring processing have transited the system, the appropriate statistics are calculated and then printed as output. If all plates required have not been processed, though, the updating of the remaining service station times is accomplished, beginning at statement C.

The shifting of program control to statement C is also effected if at statement D it is determined that no plate has completed processing at the final service station.

After updating service times, a check is made to determine whether or not another unit of throughput has arrived, and so the transfer is made to the appropriate point within statement A, thus beginning the next iteration of plate movement within the system.

The computer simulation of the steel processing facility was accomplished at M.I.T. using the Control Program -67/ Cambridge Monitor System (CP-67/CMS), which is a general purpose time-sharing system developed for use on the IBM System/360.

In an attempt to use meaningful statement numbers in the computer program, the following two categories were established:

1) Statement numbers of 1xx, 2xx, ..., 5xx have been used

in processes 1, 2, ..., 5 respectively, and enable one to quickly code those portions of the program that correspond to the individual building block modules, once the computer flow chart of the building block model is avalable.

2) Statement numbers of 1xxx and 2xxx correspond to those statements that are used for the control of those portions of the program that are not inherent in the building block programs. Thus these provide a standard means of controlling reading, writing, generation of statistics, etc.

Referring to the annotated source program included in this appendix, one can see embedded there the elements that have been described throughout this thesis and in the macro flow chart on the preceding figure. Nevertheless, it seems appropriate to comment and present reasons for some of the details of the program.

After the first few comment cards is located the type statement declaring as integers almost all the program variables. (It will be recalled that this was necessary when using fixed time increment simulation methods.) The few variables that are not specified as integers are 1) the parameters that are required to describe the appropriate stochastic process, 2) the subroutine output arguments which represent a random number, a service time, or an arrival

time, and 3) the variables that represent computed statistics, i.e., WT, EWT, etc.

The next identifiable block of statements begins with the comment card INITIALIZATION OF DATA and includes all data statements. It is noted that as a result of using CP-67/CMS it was more convenient to enter as data various items such as distribution parameters and number of plates to be processed, rather than using the procedure of reading these off computer cards. In actual practice one would normally resort to a standard read statement so that different data can be entered without the necessity of making changes to the main program itself, as is presently required.

A pair of write statements then follow, the purpose of which is to prescribe the column headings for the output which is generated for each plate upon completion of processing at the last service station, here the cutting process.

Following the write statements is a comment card that indicates the processing of a specified number of plates is about to begin. Statement number 2002 is the point at which the process of plate selection begins. After this selection, the plate is placed on the first conveyor position in the system (in this case C1PL1) by means of statement 1003.

One sees in the next block of statements, which is

headed by the comment GENERATE ARRIVAL TIME OF NEXT PLATE, the necessary conversion of the real variable output of the called subroutine. In this instance, TA, the real variable representing arrival time is converted to the corresponding integer variable, AT, by use of the FORTRAN library function subprogram, IFIX.

Beginning with the comment MOVE PLATES THROUGH CUTTING PROCESS (process 5), the movement of plates through the system in the reverse order of process number is next accomplished. It is noted that the drying machine and painting machine are referred to as "PROCESS 4" and "PROCESS 3", respectively. This is done in order to conform to the statement numbering convention that would be used if these processes contained all the elements of the realistic building block modules, namely, a conveyor, buffer storage, and machine.

An integer variable ITEM is defined as equal to the variable representing the plate size. This is done because of a FORTRAN programming restriction that forbids the use of a subscripted variable in the use of a computed go to statement.

Statement 1004 reflects the updating of the clock. Then the service times for the processes are revised. The write statement in process 5 provides instructions to print the number of plates processed and the time of completion of the plate processing. Statement 1020 performs the comparison

of SUMAT and CLOCK to determine if another unit of input is scheduled to arrive.

The if statement following 2000 CONTINUE compares the number of plates processed PP(5), to the total number of plates required to be processed NP. If the required number of plates have been processed, the statistics are generated (note the conversion of integer variables to real variables by the use of the FORTRAN library function FLOAT( ) to avoid the consequences of integer division in the computer) and the program is terminated. Otherwise, if more plates are to be processed, control is shifted to statement 1021 if no new plate arrival is scheduled, or to statement 2002 if an arrival is to take place, thus beginning the next iteration of the process.

```
C  'SIMULATION OF A VARIABLE THROUGHPUT PRODUCTION PROCESS                    SIM00010
C   STEEL PROCESSING FACILITY                                                SIM00020
C                                                                            SIM00030
      INTEGER AT, SUMAT, BUSY1, BUSY2, BUSY3, BUSY4, BUSY5, WL1, WL2,        SIM00040
     AWL3, WL4, WL5, TWT(5), TIDT(5), PP(5), C1PL1, C1PL2, C1PL3, C2PL1,SIM00050
     BC2PL2, C3PL1, C3PL2, C3PL3, C5PL1, C5PL2, HALT1, HALT2, HALT3,         SIM00060
     CHALT4, HALT5, ST1, ST2, ST3, ST4, ST5, CLOCK, DELAY(5), PL(100),       SIM00070
     DSTCAP1, STCAP2, STCAP3, STCAP4, STCAP5, PN                             SIM00080
C                                                                            SIM00090
C   INITIALIZATION OF DATA                                                   SIM00100
                                                                             SIM00110
      DATA STCAP1, STCAP2, STCAP3, STCAP4, STCAP5/ 2, 4, 0, 0, 40/           SIM00120
      DATA AT, SUMAT, BUSY1, BUSY2, BUSY3, BUSY4, BUSY5,WL1, WL2, WL3,       SIM00130
     AWL4, WL5, C1PL1, C1PL2, C1PL3, C2PL1, C2PL2, C3PL1, C3PL2,C3PL3,       SIM00140
     BC5PL1, C5PL2, HALT1, HALT2, HALT3, HALT4, HALT5, ST1, ST2, ST3,        SIM00150
     CST4, ST5, CLOCK / 33[illegible] /                                      SIM00160
      DATA TWT/5[illegible]/                                                 SIM00170
      DATA TIDT/5[illegible]/                                                SIM00180
      DATA PP/5[illegible]/                                                  SIM00190
      DATA DELAY/5[illegible]/                                               SIM00200
C   INITIAL VALUES OF ENTERING ARGUMENTS FOR RANDOM NUMBER GENERATOR         SIM00210
      DATA IAT, IP, IST11, IST12, IST13, IST2, IST51, IST52, IST53/6121,SIM00220
     A321671, 2[illegible]563, 791435, 616187,2031, 261983,647881,981635/    SIM00230
C   ALL COUNTERS SET TO ZERO AND NUMBER OF PROCESSES (PN) SPECIFIED          SIM00240
      DATA NP1, NP2, NP3, NP4, NP5, PN / 5[illegible], 5 /                   SIM00250
C     NUMBER (NP) OF PLATES TO BE COMPLETELY PROCESSED                       SIM00260
      DATA NP / 19 /                                                         SIM00270
C   PARAMETERS FOR STOCHASTIC DISTRIBUTIONS ENTERED                          SIM00280
      DATA EX2,STDX2,EX51,EX52,EX53,STDX51,STDX52,STDX53,A,B/1.5,            SIM00290
     A).15, 4.5, 3.0, 4.5, 1.5, 0.7, .25, 2.0, 1[illegible].0/               SIM00300
      WRITE(6, 1[illegible])                                                 SIM00310
      WRITE(6, 1[illegible])                                                 SIM00320
C   BEGIN PROCESSING SPECIFIED NUMBER (NP) OF PLATES                         SIM00330
      DO 2[illegible] I = 1, 5[illegible]                                    SIM00340
C   SELECT PLATE                                                             SIM00350
      CALL RANDU(IP, IY, RANUM)                                              SIM00360
```

```
      IP = IY                                                          SIM00370
      IF(RANUM .LE. 0.20) GO TO 1001                                   SIM00380
      IF(RANUM .LE. 0.45) GO TO 1002                                   SIM00390
C  ONE INCH PLATE                                                      SIM00400
      PL(I) = 3                                                        SIM00410
      GO TO 1003                                                       SIM00420
C  THREE QUARTER INCH PLATE                                            SIM00430
1002  PL(I) = 2                                                        SIM00440
      GO TO 1003                                                       SIM00450
C  HALF INCH PLATE                                                     SIM00460
1001  PL(I) = 1                                                        SIM00470
C                                                                      SIM00480
C  PLATE ENTERS SYSTEM                                                 SIM00490
1003  CIPL1 = 1                                                        SIM00500
C                                                                      SIM00510
C  GENERATE ARRIVAL TIME OF NEXT PLATE                                 SIM00520
      CALL UNIFRM(A, B, IAT, IY, TA)                                   SIM00530
      IAT = IY                                                         SIM00540
      AT = IFIX(TA)                                                    SIM00550
      SUMAT = SUMAT + AT                                               SIM00560
C  MOVE PLATES THROUGH CUTTING PROCESS (PROCESS 5)                     SIM00570
C  START CHECKING CUTTING MACHINE                                      SIM00580
 500  IF(BUSY5 .EQ. 0) GO TO 501                                       SIM00590
      IF(WL5   .EQ. 0) GO TO 502                                       SIM00600
 508  TWT(5) = TWT(5) + WL5                                            SIM00610
      GO TO 502                                                        SIM00620
 501  IF(WL5   .EQ. 0) GO TO 503                                       SIM00630
      WL5 = WL5 - 1                                                    SIM00640
 510  NP5 = NP5 + 1                                                    SIM00650
C                                                                      SIM00660
C  GENERATE SERVICE TIME                                               SIM00670
      ITEM = PL(NP5)                                                   SIM00680
      GO TO (504, 505, 506), ITEM                                      SIM00690
C  QUARTER INCH PLATE                                                  SIM00700
 504  CALL NORMAL(EX51, STDX51, IST51, IY, TS)                         SIM00710
      IST51 = IY                                                       SIM00720
```

```
      ST5 = IFIX(TS)                                                    SIM00730
      GO TO 507                                                         SIM00740
C  HALF INCH PLATE                                                     SIM00750
 505  CALL NORMAL(EX52, STDX52, IST52, IY, TS)                          SIM00760
      IST52 = IY                                                        SIM00770
      ST5 = IFIX(TS)                                                    SIM00780
      GO TO 507                                                         SIM00790
C  ONE INCH PLATE                                                      SIM00800
 506  CALL NORMAL(EX53, STDX53, IST53, IY, TS)                          SIM00810
      IST53 = IY                                                        SIM00820
      ST5 = IFIX(TS)                                                    SIM00830
 507  BUSY5 = 1                                                         SIM00840
      GO TO 508                                                         SIM00850
 508  IF(C5PL2 .EQ. 0) GO TO 509                                        SIM00860
      C5PL2 = 0                                                         SIM00870
      GO TO 510                                                         SIM00880
 509  TIDT(5) = TIDT(5) + 1                                             SIM00890
C                                                                       SIM00900
C  START CHECKING CONVEYOR 5                                           SIM00910
 502  IF(C5PL2 .EQ. 0) GO TO 511                                        SIM00920
      IF(STCAP5 - WL5) 512, 512, 513                                    SIM00930
 512  TWT(5) = TWT(5) + 1                                               SIM00940
      IF(C5PL1 .EQ. 0) GO TO 400                                        SIM00950
      TWT(5) = TWT(5) + 1                                               SIM00960
      HALT5 = 1                                                         SIM00970
      GO TO 400                                                         SIM00980
 513  WL5 = WL5 + 1                                                     SIM00990
      C5PL2 = 0                                                         SIM01000
 511  IF(C5PL1 .EQ. 0) GO TO 400                                        SIM01010
      C5PL2 = 1                                                         SIM01020
      C5PL1 = 0                                                         SIM01030
C                                                                       SIM01040
C  MOVE PLATES THROUGH COATING PROCESS  (PROCESSES 4 & 3)               SIM01050
C  CHECK DRYING MACHINE    ("PROCESS 4")                               SIM01060
 401  IF(BUSY4 .EQ. 0) GO TO 301                                        SIM01070
C  CHECK PAINTING MACHINE    ("PROCESS 3")                             SIM01080
```

```
      IF(BUSY3 .EQ. 0) GO TO 302                                        SIM01090
      TWT(3) = TWT(3) + 1                                               SIM01100
      IF(C3PL3 .EQ. 0) GO TO 303                                        SIM01110
      TWT(3) = TWT(3) + 1                                               SIM01120
      IF(C3PL2 .EQ. 0) GO TO 304                                        SIM01130
      TWT(3) = TWT(3) + 1                                               SIM01140
      IF(C3PL1 .EQ. 0) GO TO 200                                        SIM01150
      TWT(3) = TWT(3) + 1                                               SIM01160
      HALT3 = 1                                                         SIM01170
      GO TO 200                                                         SIM01180
  301 IF(BUSY3 .EQ. 0) GO TO 305                                        SIM01190
      BUSY4 = 1                                                         SIM01200
      BUSY3 = 0                                                         SIM01210
      GO TO 302                                                         SIM01220
  305 TIDT(4) = TIDT(4) + 1                                             SIM01230
  302 IF(C3PL3 .EQ. 0) GO TO 306                                        SIM01240
      BUSY3 = 1                                                         SIM01250
      C3PL3 = 0                                                         SIM01260
      GO TO 303                                                         SIM01270
  306 TIDT(3) = TIDT(3) + 1                                             SIM01280
  303 IF(C3PL2 .EQ. 0) GO TO 304                                        SIM01290
      C3PL3 = 1                                                         SIM01300
      C3PL2 = 0                                                         SIM01310
  304 IF(C3PL1 .EQ. 0) GO TO 200                                        SIM01320
      C3PL2 = 1                                                         SIM01330
      C3PL1 = 0                                                         SIM01340
C                                                                       SIM01350
C  MOVE PLATES THROUGH BLASTING PROCESS    (PROCESS 2)                  SIM01360
C  CHECK BLASTING MACHINE                                               SIM01370
  200 IF(BUSY2 .EQ. 0) GO TO 201                                        SIM01380
      IF(WL2   .EQ. 0) GO TO 202                                        SIM01390
  208 TWT(2) = TWT(2) + WL2                                             SIM01400
      GO TO 202                                                         SIM01410
  201 IF(WL2   .EQ. 0) GO TO 203                                        SIM01420
      WL2 = WL2 - 1                                                     SIM01430
  210 NP2 = NP2 + 1                                                     SIM01440
```

```
C                                                                       SIM01450
C  GENERATE SERVICE TIME                                                SIM01460
      CALL NORMAL(EX2, STDX2, IST2, IY, TS)                             SIM01470
      IST2 = IY                                                         SIM01480
      ST2 = IFIX(TS)                                                    SIM01490
  207 BUSY2 = 1                                                         SIM01500
      GO TO 208                                                         SIM01510
  203 IF(C2PL2 .EQ. 0) GO TO 209                                        SIM01520
      C2PL2 = 0                                                         SIM01530
      GO TO 210                                                         SIM01540
  209 TIDT(2) = TIDT(2) + 1                                             SIM01550
C  START CHECKING CONVEYOR 2                                            SIM01560
  202 IF(C2PL2 .EQ. 0) GO TO 211                                        SIM01570
      IF(STCAP2 - WL2) 212, 212, 213                                    SIM01580
  212 TWT(2) = TWT(2) + 1                                               SIM01590
      IF(C2PL1 .EQ. 0) GO TO 100                                        SIM01600
      TWT(2) = TWT(2) + 1                                               SIM01610
      HALT2 = 1                                                         SIM01620
      GO TO 100                                                         SIM01630
  213 WL2 = WL2 + 1                                                     SIM01640
      C2PL2 = 0                                                         SIM01650
  211 IF(C2PL1 .EQ. 0) GO TO 100                                        SIM01660
      C2PL2 = 1                                                         SIM01670
      C2PL1 = 0                                                         SIM01680
C                                                                       SIM01690
C  MOVE PLATES THROUGH LEVELING PROCESS   (PROCESS 1)                   SIM01700
C  CHECK LEVELING MACHINE                                               SIM01710
  101 IF(BUSY1 .EQ. 0) GO TO 101                                        SIM01720
      IF(WL1   .EQ. 0) GO TO 102                                        SIM01730
  108 TWT(1) = TWT(1) + WL1                                             SIM01740
      GO TO 102                                                         SIM01750
  111 IF(WL1   .EQ. 0) GO TO 103                                        SIM01760
      WL1 = WL1 - 1                                                     SIM01770
  110 NP1 = NP1 + 1                                                     SIM01780
C  GENERATE SERVICE TIME                                                SIM01790
      ITEM = PL(NP1)                                                    SIM01800
```

```
      GO TO (104, 105, 106), ITEM                                    SIM01810
C  HALF INCH PLATE                                                   SIM01820
 104  CALL RANDU(IST11, IY, R)                                       SIM01830
      IST11 = IY                                                     SIM01840
      IF(R .LE. 0.10) GO TO 114                                      SIM01850
      ST1 = 1                                                        SIM01860
      GO TO 107                                                      SIM01870
 114  ST1 = 2                                                        SIM01880
      GO TO 107                                                      SIM01890
C  THREE QUARTER INCH PLATE                                          SIM01900
 105  CALL RANDU(IST12, IY, R)                                       SIM01910
      IST12 = IY                                                     SIM01920
      IF(R .LE. 0.20) GO TO 115                                      SIM01930
      ST1 = 1                                                        SIM01940
      GO TO 107                                                      SIM01950
 115  ST1 = 2                                                        SIM01960
C  ONE INCH PLATE                                                    SIM01970
 106  CALL RANDU(IST13, IY, R)                                       SIM01980
      IST13 = IY                                                     SIM01990
      IF(R .LE. 0.30) GO TO 116                                      SIM02000
      IF(R .LE. 0.90) GO TO 117                                      SIM02010
      ST1 = 3                                                        SIM02020
      GO TO 107                                                      SIM02030
 117  ST1 = 2                                                        SIM02040
      GO TO 107                                                      SIM02050
 116  ST1 = 1                                                        SIM02060
 107  BUSY1 = 1                                                      SIM02070
      GO TO 108                                                      SIM02080
 103  IF(C1PL3 .EQ. 0) GO TO 109                                     SIM02090
      C1PL3 = 0                                                      SIM02100
      GO TO 110                                                      SIM02110
 109  TIDT(1) = TIDT(1) + 1                                          SIM02120
C                                                                    SIM02130
C  START CHECKING CONVEYOR ONE                                       SIM02140
 102  IF(C1PL3 .EQ. 0) GO TO 111                                     SIM02150
      IF(STCAP1 - WL1) 112,112, 113                                  SIM02160
```

```
 112  TWT(1) = TWT(1) + 1                                              SIM02170
      IF(C1PL2 .EQ. 0) GO TO 118                                       SIM02180
      TWT(1) = TWT(1) +1                                               SIM02190
      IF(C1PL1 .EQ. 0) GO TO 1004                                      SIM02200
      TWT(1) = TWT(1) + 1                                              SIM02210
      HALT1 = 1                                                        SIM02220
      GO TO 1004                                                       SIM02230
 113  WL1 = WL1 + 1                                                    SIM02240
      C1PL3 = 0                                                        SIM02250
 111  IF(C1PL2 .EQ. 0) GO TO 118                                       SIM02260
      C1PL3 = 1                                                        SIM02270
      C1PL2 = 0                                                        SIM02280
 118  IF(C1PL1 .EQ. 0) GO TO 1004                                      SIM02290
      C1PL2 = 1                                                        SIM02300
      C1PL1 = 0                                                        SIM02310
C                                                                      SIM02320
C  UPDATE CLOCK                                                        SIM02330
1004  CLOCK = CLOCK + 1                                               SIM02340
C                                                                      SIM02350
C  REVISE SERVICE TIME FOR CUTTING PROCESS                             SIM02360
 520  IF(ST5 - 1) 420, 521, 522                                        SIM02370
 521  ST5 = ST5 - 1                                                    SIM02380
      BUSY5 = 0                                                        SIM02390
      PP(5) = PP(5) + 1                                                SIM02400
      WRITE(6, 1005) PP(5), CLOCK                                      SIM02410
      GO TO 420                                                        SIM02420
 522  ST5 = ST5 - 1                                                    SIM02430
C  REVISE SERVICE TIME OF COATING PROCESS                              SIM02440
 421  IF(ST4 - 1) 423, 421, 422                                        SIM02450
 423  IF(BUSY4 .EQ. 0) GO TO 320                                       SIM02460
 425  IF(HALT5 .EQ. 0) GO TO 424                                       SIM02470
      DELAY(5) = DELAY(5) + 1                                          SIM02480
      HALT5 = 0                                                        SIM02490
      GO TO 320                                                        SIM02500
 424  BUSY4 = 0                                                        SIM02510
      PP(4) = PP(4) + 1                                                SIM02520
```

```
      C5PL1 = 1                                                    SIM02530
      GO TO 329                                                    SIM02540
  421 ST4 = ST4 - 1                                                SIM02550
      GO TO 425                                                    SIM02560
  422 ST4 = ST4 - 1                                                SIM02570
C                                                                  SIM02580
C  REVISE SERVICE TIME  OF PAINTING MACHINE                        SIM02590
  32' IF(ST3 - 1) 323, 321, 322                                    SIM02600
 323  IF(BUSY3 .EQ. 0) GO TO 220                                   SIM02610
  325 IF(HALT4 .EQ. 0) GO TO 324                                   SIM02620
      DELAY(4) = DELAY(4) + 1                                      SIM02630
      HALT4 = 0                                                    SIM02640
      GO TO 220                                                    SIM02650
  324 BUSY3 = 0                                                    SIM02660
      PP(3) = PP(3) + 1                                            SIM02670
      BUSY4 = 1                                                    SIM02680
      GO TO 220                                                    SIM02690
  321 ST3 = ST3 - 1                                                SIM02700
      GO TO 325                                                    SIM02710
  322 ST3 = ST3 - 1                                                SIM02720
C                                                                  SIM02730
C  REVISE SERVICE TIME OF BLASTING MACHINE                         SIM02740
  22  IF(ST2 - 1) 223, 221, 222                                    SIM02750
  223 IF(BUSY2 .EQ. 0) GO TO 120                                   SIM02760
  225 IF(HALT3 .EQ. 0) GO TO 224                                   SIM02770
      DELAY(3) = DELAY(3) + 1                                      SIM02780
      HALT3 = 0                                                    SIM02790
  224 BUSY2 = 0                                                    SIM02800
      PP(2) = PP(2) + 1                                            SIM02810
      C3PL1 = 1                                                    SIM02820
      GO TO 120                                                    SIM02830
  221 ST2 = ST2 - 1                                                SIM02840
      GO TO 225                                                    SIM02850
  222 ST2 = ST2 - 1                                                SIM02860
C                                                                  SIM02870
C  REVISE LEVELER SERVICE TIME                                     SIM02880
```

```
 12   IF(ST1 - 1) 123, 121, 122                                          SIM02890
 123  IF(BUSY1 .EQ. 0) GO TO 1020                                        SIM02900
 125  IF(HALT2 .EQ. 0) GO TO 124                                         SIM02910
      DELAY(2) = DELAY(2) + 1                                            SIM02920
      HALT2 = 0                                                          SIM02930
      GO TO 1020                                                         SIM02940
 124  BUSY1 = 0                                                          SIM02950
      PP(1) = PP(1) + 1                                                  SIM02960
      C2PL1 = 1                                                          SIM02970
      GO TO 1020                                                         SIM02980
 121  ST1 = ST1 - 1                                                      SIM02990
      GO TO 125                                                          SIM03000
 122  ST1 = ST1 -1                                                       SIM03010
1020  IF(PP(5) .EQ. NP) GO TO 2001                                       SIM03020
C                                                                        SIM03030
C  COMPARE CLOCK AND SUMAT TO SEE IF ANOTHER ARRIVAL IS DUE              SIM03040
      IF(CLOCK .LT. SUMAT) GO TO 500                                     SIM03050
      IF(HALT1 .EQ. 0) GO TO 2000                                        SIM03060
      SUMAT = SUMAT + 1                                                  SIM03070
      DELAY(1) = DELAY(1) + 1                                            SIM03080
      HALT1 = 0                                                          SIM03090
      GO TO 500                                                          SIM03100
2000  CONTINUE                                                           SIM03110
C                                                                        SIM03120
C  GENERATE STATISTICS                                                   SIM03130
2001  WRITE(6, 1007)                                                     SIM03140
      WRITE(6, 1008)                                                     SIM03150
      DO 1030 IS = 1, PN                                                 SIM03160
C                                                                        SIM03170
C  EXPECTED WAITING TIME                                                 SIM03180
      WT = FLOAT(TWT(IS))                                                SIM03190
      EWT = WT / PP(IS)                                                  SIM03200
C                                                                        SIM03210
C  EXPECTED IDLE TIME OF PROCESS                                         SIM03220
      TID = FLOAT(TIDT(IS))                                              SIM03230
      EIDT = TID / PP(IS)                                                SIM03240
```

```
      WRITE(6, 1006) IS, PP(IS), TWT(IS),EWT, TIDT(IS),EIDT, DELAY(IS)   SIM03250
1030  CONTINUE                                                           SIM03260
1005  FORMAT(3(3X, I3, 3X, I5, 4X))                                      SIM03270
1006  FORMAT(3X,I2,6X,I3,5X,I5,1X,F5.1,3X,I5,1X,F5.1,3X,I5)              SIM03280
1008  FORMAT(3X,'NO.  /PROCESSED/TOTAL/EXPECTED/TOTAL/EXPECTED/')        SIM03290
1007  FORMAT(/1X,'PROCESS   PLATES   WAITING TIME    IDLE TIME    DELAY'SIM03300
     A)                                                                  SIM03310
1009  FORMAT(1X'PLATE COMPLETED/ PLATE COMPLETED/ PLATE COMPLETED/')     SIM03320
1011  FORMAT(2X'NO.   AT TIME /  NO.   AT TIME /  NO.   AT TIME /')      SIM03330
      CALL EXIT                                                          SIM03340
      END                                                                SIM03350
```

```
l novak
      ENTER PASSWORD:

      READY AT 12.34.53 ON 05/17/71


          ipl cms
          CMS..VERSION 3.0-1 03/01/71

          $ simprod fortran
          EXECUTION BEGINS...
          PLATE COMPLETED/ PLATE COMPLETED/ PLATE COMPLETED/
           NO.   AT TIME /  NO.   AT TIME /  NO.   AT TIME /
              1       16
              2       20
              3       30
              4       35
              5       40
              6       43
              7       47
              8       51
              9       54
             10       64
             11       68
             12       70
             13       73
             14       77
             15       82

          PROCESS   PLATES   WAITING TIME     IDLE TIME     DELAY
            NO.  /PROCESSED/TOTAL/EXPECTED/TOTAL/EXPECTED/
              1       18          0   0.0       52   2.9        0
              2       17          0   0.0       62   3.6        0
              3       17          0   0.0       65   3.8        0
              4       17          0   0.0       65   3.8        0
              5       15        153  10.2       11   0.7        0
          R; T=0.63/2.07 12.36.56
```

Bibliography

[1] Allen, M., "The Efficient Utilization of Labor under Conditions of Fluctuating Demand", Industrial Scheduling, C. XVI (Eds. J. F. Muth and G. L. Thompson), Englwood Cliffs, N. J.: Prentice Hall, Inc., 1963.

[2] Baker, C. T., Dzielinski, B. P., "Simulation of a Simplified Job Shop", Management Science, Vol. VI, 1960, pp. 311-323.

[3] Bellman, R. E., Adaptive Control Processes, Princeton University Press, 1961.

[4] Beusch, John Ulrich, "Dynamic Behavior and Control of Communication Networks", (Unpublished Ph.D. Thesis, May, 1965, M.I.T.)

[5] Buffa, Elwood S., Production Inventory Systems, Homeward, Illinois: Richard D. Irwin, Inc., 1968.

[6] Buyron, C. D., Introduction to the Mathematical Theory of Control Processes, New York: Academic Press, 1967.

[7] Carroll, D. C., "Heuristic Sequencing of Single and Multiple Component Jobs", (Unpublished Ph.D. Dissertation, Sloan School of Management, M.I.T., 1965).

[8] Chu, Kong, and Naylor, T. H., "Two Alternative Methods for Simulating Waiting Line Models", Journal of Industrial Engineering, Nov.-Dec., 1965.

[9] Churchman, Arkoff, Arnoff. Introduction to Operations Research, New York: John Wiley and Sons, Inc., 1957.

[10] Churchman, C. West, et al. Introduction to Operations Research, London: John Wiley and Sons, Inc., 1957.

[11] Conway, R. W., Johnson, B. M. and Maxwell, W. L., "Some Problems of Digital Systems Simulations", Management Science, October 1959.

[12] Conway, R. W. and Maxwell, W. L., "Network Scheduling by the Shortest Operations Discipline", Operations Research, Vol. 10, No. 1, 1962, pp. 51-73.

[13] Conway, R. W., Maxwell, W. L. and Miller, L. W. Theory of Scheduling, Reading, Mass.: Addison-Wesley Publishing Co., 1967.

[14] Dantzig, G. B., "A Machine-Job Scheduling Model", Management Science, Vol. VI, 1960, pp. 191-196.

[15] Derman, Cyrus, "Markovian Sequential Control Processes Denumerable State Space", Journal of Mathematical Analysis and Applications, Vol. 10, 1965, pp. 295-302.

[16] Devanney, J. W., "Permutation Schedules for the n/m Job Shop Problem", (Unpublished paper, 1969).

[17] Drake, Alvin W., Fundamentals of Applied Probability Theory. New York: McGraw-Hill Book Co., 1967.

[18] Elmaghraby, Salah E. The Design of Production Systems. New York: Reinhold Publishing Corp., 1966.

[19] Feller, William. An Introduction to Probability Theory and Its Applications, Vol. II. New York; John Wiley and Sons, Inc., 1966.

[20] Frankel, E. G. "Ship Production Scheduling and Control", (Unpublished Notes, M.I.T., 1970).

[21] International Business Machines Corporation. General Purpose Systems Simulator II, Reference Manual.

[22] Greene, James H. Production Control Systems and Decisions. Homewood, Illinois: Richard D. Irwin, Inc., 1965.

[23] Harris, R. D., "An Empirical Investigation and Model Proposal of a Job Shop-Like Queueing System", Western Management Science Institute, Working Paper No. 84, U.C.L.A., July 1965.

[24] Hadley, G. Linear Programming. Reading, Mass.: Addison-Wesley Publishing Co., Inc., 1962.

[25] Jackson, James R., "Jobshop-like Queueing Systems", Management Science, Vol. 10, 1963, pp. 131-142.

[26] Jackson, James R., "Multiple Servers with Limited Waiting Space", Naval Research Logistics Quarterly, Vol. 5, 1958, pp. 315-321.

[27] Kiviat, Philip J. and Colker, Alan, "GASP - a General Activity Simulation Program", The Rand Corporation, P-286A, February 1964.

[28] Klein, Morton, "Some Production Planning Problems", Naval Research Logistics Quarterly, Vol. 4, 1957, pp. 269-286.

[29] Le Grande, E., "The Development of a Factory Simulation System Using Actual Operating Data", Management Technology, Vol. 3, Vol. 1, 1963.

[30] Markowitz, H. M. et al, "SIMSCRIPT: A Simulation Programming Language," The Rand Corporation, RM-3310, November 1962.

[31] International Business Machines Corporation. Mathematical Programming System/360, User's Manual.

[32] McMillan, Claude and Gonzalez, Richard F. Systems Analysis - A Computer Approach to Decision Models. Homewood, Illinois: Richard D. Irwin, Inc., 1965 and 1968.

[33] Moder, Joseph J. and Phillips, Cecil R. Jr., "Queueing with Fixed and Variable Channels", Operations Research, Vol. 16, 1968, pp. 218-231.

[34] Muller, Mervin E., "A Comparison of Methods for Generating Normal Deviates on Digital Computers", Journal of the Association for Computing Machinery, VI, 1959, pp. 376-383.

[35] Naylor, Thomas H. et al, Computer Simulation Techniques. New York: John Wiley and Sons., Inc., 1966.

[36] Nelson, R. T., "Labor and Machine Limited Production Systems", Management Science, Vol. 13, No. 9, May 1967, pp. 648-71.

[37] Nelson, Rosser T., "Queueing Network Experiments with Varying Arrival and Service Processes", Naval Research Logistics Quarterly, Vol. 13, 1966, pp. 321-347.

[38] Nemhauser, George L. Introduction to Dynamic Programming. New York: John Wiley and Sons., Inc., 1966.

[39] Newcomb, R. W. Concepts of Linear Systems and Controls. Belmont, California: Brooks/Cole Publishing Co., 1968.

[40] Prabhu, N. V. Queues and Inventories. New York: John Wiley and Sons, Inc., 1966.

[41] Pugh, Alexander L. Dynamo User's Manual. Cambridge, Mass.: The M.I.T. Press, 1963.

[42] Report of System Simulation Symposium, 8th National Convention of the American Institute of Industrial Engineers, New York, May 16, 1957.

[43] Riorden, John. Stochastic Service Systems. New York: John Wiley and Sons., Inc., 1962.

[44] Rowe, A. J., "Application of Computer Simulation to Sequential Decision Rules in Production Scheduling", Proceedings: 11th Annual Industrial Engineering Institute, University of California, Berkeley-Los Angeles, February 1959.

[45] Rowe, A. J., "Towards the Theory of Scheduling", Industrial Engineering, Vol. 11, March 1960, pp. 125-136.

[46] Saaty, T. L. Elements of Queueing Theory with Applications. New York: McGraw-Hill, 1961.

[47] Saaty, T. L. Mathematical Models of Operations Research. New York: McGraw-Hill, 1959.

[48] Saunders, L. R., "Probability Functions for Waiting Times in Single-Channel Queues, with Emphasis on Simple Approximations", Operations Research, Vol. 9, 1961, pp. 351-362.

[49] Scientific Subroutine Packages, Reference Manual, International Business Machines Corporation.

[50] Smith, John. Computer Simulation Models. New York: Hafner Publishing Co., 1968.

[51] Williams, J. W. J, "The Elliot Simulation Package", The Computer Journal, VI, January 1964, pp. 328-331.